



AMERICAN INTERNATIONAL UNIVERSITY–BANGLADESH (AIUB)
FACULTY OF SCIENCE & TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
INTRODUCTION TO DATABASE
Spring 2022-2023
Section: M, Group: 01

PROJECT ON
Air Ticket Booking & Boarding Management System

Supervised By
KAWSER IROM RUSHEE

Submitted By

Name	ID	Contribution
1. NILOY, NAFIUR RAHMAN	22-46459-1	
2. RAHMAN, AZMINUR	22-46588-1	
3. KUNDU, SAIKOT	22-46615-1	
4. SARWAR, MD. SAKIB	22-46625-1	

Date of Submission: May 17, 2023

TABLE OF CONTENTS

SL No	Topic Name	Page No
01.	Cover Page	01
02.	Table of Contents	02
03.	Introduction	03
04.	Scenario	03
05.	ER-DIAGRAM (Screenshot)	04
06.	ER-DIAGRAM (Final)	05
07.	Normalization	06-10
08.	Final Tables	10
09.	Table Creation	10-12
10.	Table Creation (Screenshot)	12-16
11.	Data Insertion	17-19
12.	Joining	20-21
13.	Subquery	22-23
14.	View	24
15.	Add Constraint	25

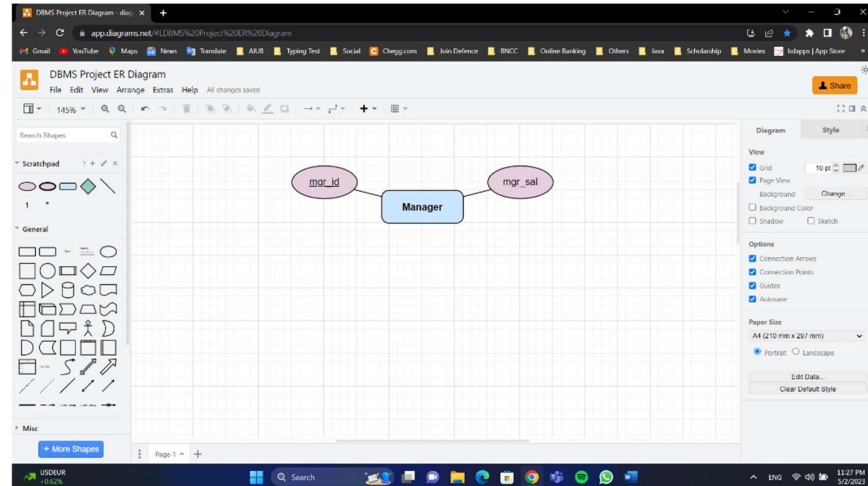
**Introduction:**

'Uran' is a company that operates an air ticket booking and boarding management system. The system is managed by managers who oversee ticket agents. Ticket agents sell tickets to customers. Customers can buy tickets for themselves or for other passengers. Passenger must need ticket for boarding in planes. The company also store the route of planes.

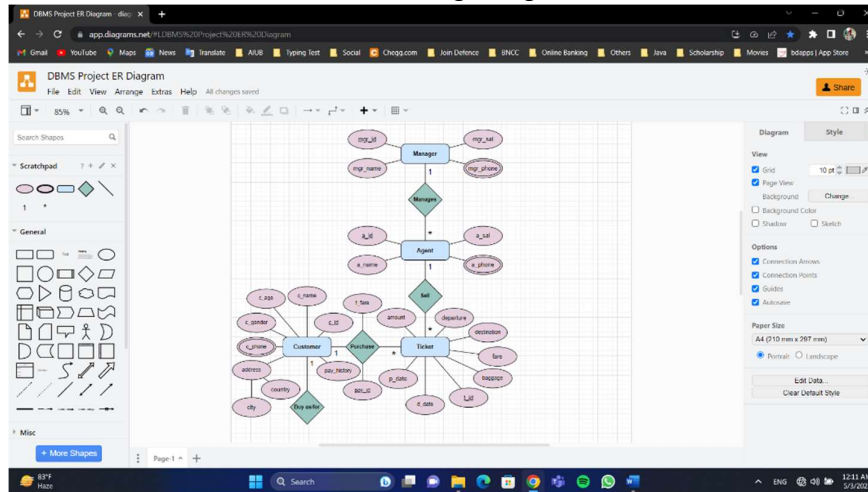
Scenario:

The owners of an air ticket booking and boarding management system 'Uran' wants to create a database management system for their company. In this company A manager can **manage** many ticket agent. Managers can be identified using their unique manager ID. Each manager's name, phone number, and salary are stored in the database. One or more ticket agent may work under a manager. A ticket agent can **sell** many ticket. Every agent's name, phone, salary, as well as a unique agent ID is recorded. Tickets have their fare, amount, departure, destination, departure date, purchase date and baggage stored as records along with a unique ticket ID for ticket. A customer must **purchase** ticket for traveling. During purchasing ticket payment id and total fare are also stored in the database. Customers are identified by their unique Customer ID. Aside from that, a customer's name, address with city & country, phone number, age, gender, and payment history (if available) are also recorded in the database. A customer may be a passenger themselves or **buy as/for** tickets for another passenger. Passengers can be distinctly identified with the help of their Passenger ID. Additionally, a passenger's name, address with city & country, age, phone number, and gender are also stored within the database. Passengers need tickets for traveling. Passengers will use their tickets for **boarding** to plane. Each plane has a distinct plane ID alongside its terminal number, take-off airport, take-off time, destination and capacity. Every passenger **sits on** a seat. Every seat identified by unique seat id alongside seat type. A plane has many seats. A plane **travel in** one route towards their destination. These routes are distinctly marked using a route ID and it's had distance.

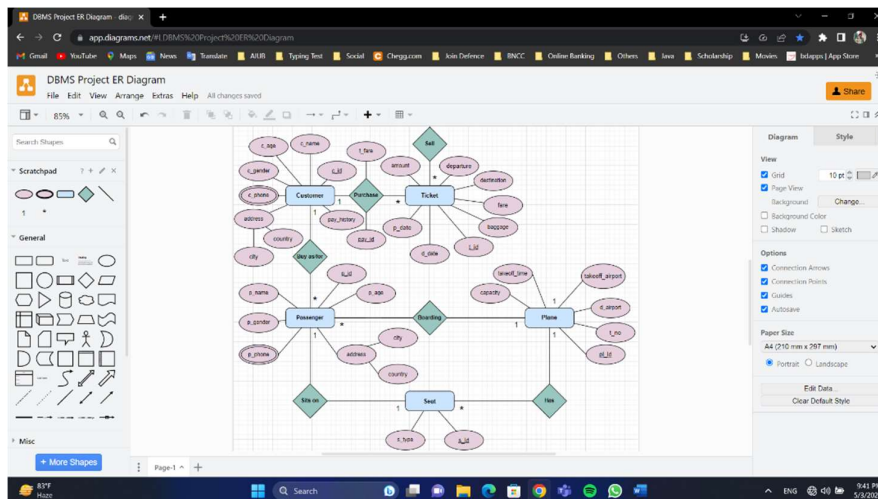
ER-DIAGRAM (Screenshot):



At the beginning

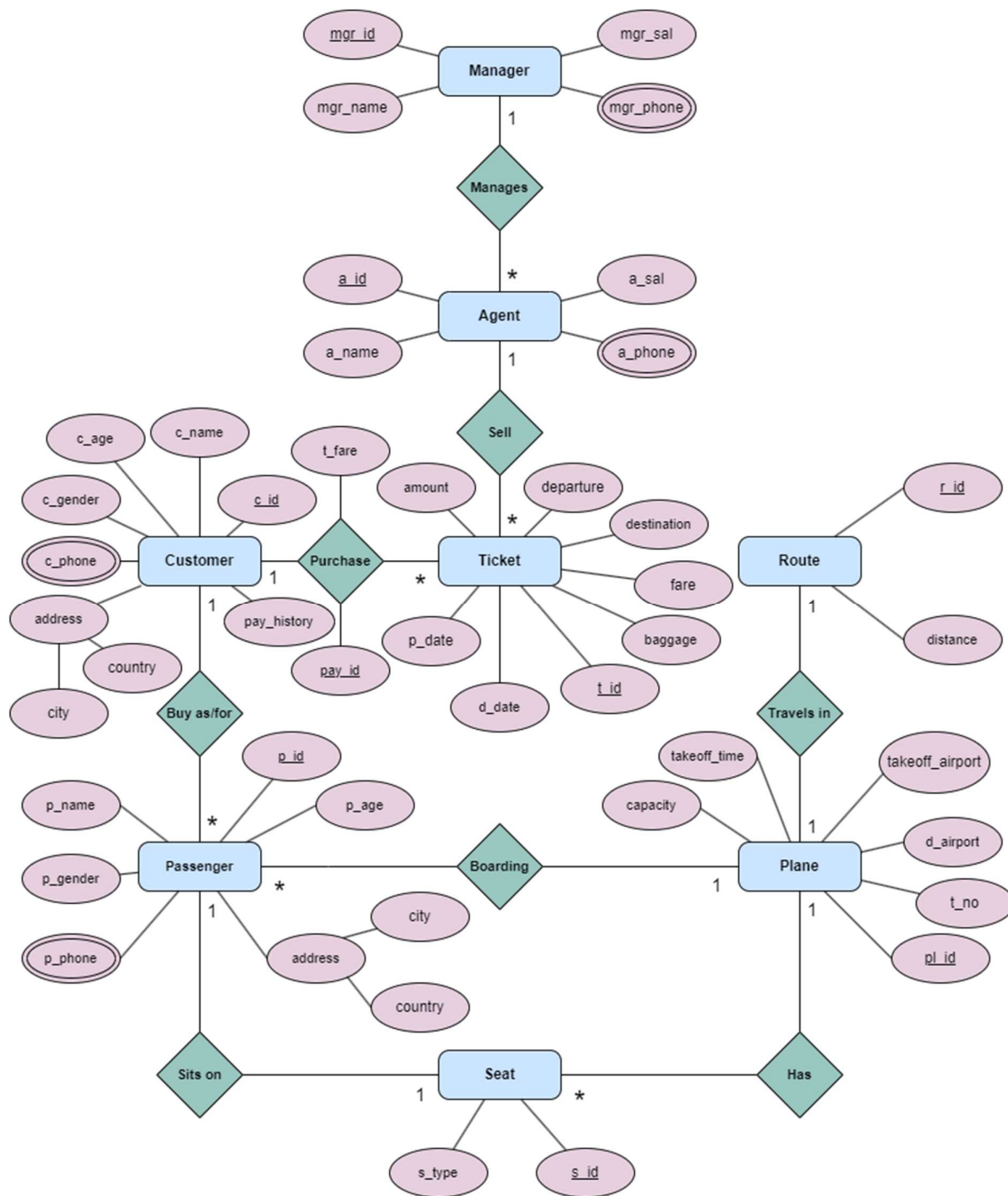


At the Middle



At nearly end

ER-DIAGRAM (Final):



ER-DIAGRAM

Normalization:

- **Manages**

UNF: mgr_id, mgr_name, mgr_sal, mgr_phone, a_id, a_name, a_sal, a_phone

1NF: (**mgr_phone** and **a_phone** is multivalued attribute)

mgr_id, mgr_name, mgr_sal, mgr_phone, a_id, a_name, a_sal, a_phone

2NF:

- I. mgr_id, mgr_name, mgr_sal
- II. mgr_phone, mgr_id (fk)
- III. a_id, a_name, a_sal, mgr_id (fk)
- IV. a_phone, a_id (fk)

3NF: (No Transitive Dependency)

- I. mgr_id, mgr_name, mgr_sal
- II. mgr_phone, mgr_id (fk)
- III. a_id, a_name, a_sal, mgr_id (fk)
- IV. a_phone, a_id (fk)

Tables:

1. mgr_id, mgr_name, mgr_sal
2. mgr_phone, mgr_id (fk)
3. a_id, a_name, a_sal, mgr_id (fk)
4. a_phone, a_id (fk)

- **Sell**

UNF: a_id, a_name, a_sal, a_phone, t_id, amount, fare, departure, destination, p_date, d_date, baggage

1NF: (**a_phone** is a multivalued attribute)

a_id, a_name, a_sal, a_phone, t_id, amount, fare, departure, destination, p_date, d_date, baggage

2NF:

- I. a_id, a_name, a_sal
- II. a_phone, a_id (fk)
- III. t_id, amount, fare, departure, destination, p_date, d_date, baggage, a_id (fk)

3NF: (No Transitive Dependency)

- I. a_id, a_name, a_sal
- II. a_phone, a_id (fk)
- III. t_id, amount, fare, departure, destination, p_date, d_date, baggage, a_id (fk)

Tables:

1. a_id, a_name, a_sal
2. a_phone, a_id (fk)
3. t_id, amount, fare, departure, destination, p_date, d_date, baggage, a_id (fk)

- **Purchase**

UNF: t_id, amount, fare, departure, destination, p_date, d_date, baggage, c_id, c_name, c_age, c_gender, city, country, c_phone, pay_history, pay_id, t_fare

1NF: (**c_phone** is a multivalued attribute)

t_id, amount, fare, departure, destination, p_date, d_date, baggage, c_id, c_name, c_age, c_gender, city, country, c_phone, pay_history, pay_id, t_fare

2NF:

- I. t_id, amount, fare, departure, destination, p_date, d_date, baggage, c_id (fk)
- II. c_id, c_name, c_age, c_gender, city, country, pay_history
- III. c_phone, c_id (fk)
- IV. pay_id, t_fare, c_id (fk), t_id (fk)

3NF: (Transitive Dependency)

- I. t_id, amount, fare, departure, destination, p_date, d_date, baggage, c_id (fk)
- II. c_id, c_name, c_age, c_gender, pay_history
- III. city, country, c_id (fk)
- IV. c_phone, c_id (fk)
- V. pay_id, t_fare, c_id (fk), t_id (fk)

Tables:

1. t_id, amount, fare, departure, destination, p_date, d_date, baggage, c_id (fk)
2. c_id, c_name, c_age, c_gender, pay_history
3. city, country, c_id (fk)
4. c_phone, c_id (fk)
5. pay_id, t_fare, c_id (fk), t_id (fk)

• **Buy As/For**

UNF: c_id, c_name, c_age, c_gender, city, country, c_phone, pay_history, p_id, p_name, p_gender, p_age, city, country, p_phone

1NF: (c_phone, p_phone is a multivalued attribute)

c_id, c_name, c_age, c_gender, city, country, c_phone, pay_history, p_id, p_name, p_gender, p_age, city, country, p_phone

2NF:

- I. c_id, c_name, c_age, c_gender, city, country, pay_history
- II. c_phone, c_id (fk)
- III. p_id, p_name, p_gender, p_age, city, country, c_id (fk)
- IV. p_phone, p_id (fk)

3NF: (Transitive Dependency)

- I. c_id, c_name, c_age, c_gender, pay_history
- II. city, country, c_id (fk)
- III. c_phone, c_id (fk)
- IV. p_id, p_name, p_gender, p_age, c_id (fk)
- V. city, country, p_id (fk)
- VI. p_phone, p_id (fk)

Table:

1. c_id, c_name, c_age, c_gender, pay_history
2. city, country, c_id (fk)
3. c_phone, c_id (fk)
4. p_id, p_name, p_gender, p_age, c_id (fk)
5. city, country, p_id (fk)
6. p_phone, p_id (fk)

- **Boarding**

UNF: p_id, p_name, p_gender, p_age, city, country, p_phone, pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity

1NF: (p_phone is a multivalued attribute)

p_id, p_name, p_gender, p_age, city, country, p_phone, pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity

2NF:

- I. p_id, p_name, p_gender, p_age, city, country, pl_id (fk)
- II. p_phone, p_id (fk)
- III. pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity

3NF: (Transitive Dependency)

- I. p_id, p_name, p_gender, p_age, pl_id (fk)
- II. city, country, p_id (fk)
- III. p_phone, p_id (fk)
- IV. pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity

Table:

1. p_id, p_name, p_gender, p_age, pl_id (fk)
2. city, country, p_id (fk)
3. p_phone, p_id (fk)
4. pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity

- **Sits On**

UNF: p_id, p_name, p_gender, p_age, city, country, p_phone, s_id, s_type

1NF: (p_phone is a multivalued attribute)

p_id, p_name, p_gender, p_age, city, country, p_phone, s_id, s_type

2NF:

- I. p_id, p_name, p_gender, p_age, city, country
- II. p_phone, p_id (fk)
- III. s_id, s_type, p_id (fk)

3NF: (Transitive Dependency)

- I. p_id, p_name, p_gender, p_age
- II. city, country, p_id (fk)
- III. p_phone, p_id (fk)
- IV. s_id, s_type, p_id (fk)

Table:

1. p_id, p_name, p_gender, p_age
2. city, country, p_id (fk)
3. p_phone, p_id (fk)
4. s_id, s_type, p_id (fk)

- **Has**

UNF: s_id, s_type, pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity

1NF: (No multivalued attribute)

s_id, s_type, pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity

2NF:

- I. s_id, s_type, pl_id (fk)
- II. pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity

3NF: (No Transitive Dependency)

- I. s_id, s_type, pl_id (fk)
- II. pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity

Table:

1. s_id, s_type, pl_id (fk)
2. pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity

• **Travels In**

UNF: pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity, r_id, distance

1NF: (No multivalued attribute)

pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity, r_id, distance

2NF:

- I. pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity
- II. r_id, distance, pl_id (fk)

3NF: (No Transitive Dependency)

- I. pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity
- II. r_id, distance, pl_id (fk)

Table:

1. pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity
2. r_id, distance, pl_id (fk)

• **Tables**

1. mgr_id, mgr_name, mgr_sal
2. mgr_phone, mgr_id (fk)
3. a_id, a_name, a_sal, mgr_id (fk)
4. a_phone, a_id (fk)
5. a_id, a_name, a_sal
6. a_phone, a_id (fk)
7. t_id, amount, fare, departure, destination, p_date, d_date, baggage, a_id (fk)
8. t_id, amount, fare, departure, destination, p_date, d_date, baggage, c_id (fk)
9. c_id, c_name, c_age, c_gender, pay_history
10. city, country, c_id (fk)
11. c_phone, c_id (fk)
12. pay_id, t_fare, c_id (fk), t_id (fk)
13. c_id, c_name, c_age, c_gender, pay_history
14. city, country, c_id (fk)
15. c_phone, c_id (fk)
16. p_id, p_name, p_gender, p_age, c_id (fk)
17. city, country, p_id (fk)
18. p_phone, p_id (fk)
19. p_id, p_name, p_gender, p_age, pl_id (fk)
20. city, country, p_id (fk)
21. p_phone, p_id (fk)
22. pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity

23. p_id, p_name, p_gender, p_age
24. city, country, p_id (fk)
25. p_phone, p_id (fk)
26. s_id, s_type, p_id (fk)
27. s_id, s_type, pl_id (fk)
28. pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity
29. pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity
30. r_id, distance, pl_id (fk)

Final Tables

1. mgr_id, mgr_name, mgr_sal
2. mgr_phone, mgr_id (fk)
3. a_id, a_name, a_sal, mgr_id (fk)
4. a_phone, a_id (fk)
5. cc_id, city, country
6. c_id, c_name, c_age, c_gender, pay_history, cc_id (fk)
7. c_phone, c_id (fk)
8. t_id, amount, fare, departure, destination, p_date, d_date, baggage, a_id (fk), c_id (fk)
9. pay_id, t_fare, c_id (fk), t_id (fk)
10. pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity
11. p_id, p_name, p_gender, p_age, c_id (fk), pl_id (fk), cc_id (fk)
12. p_phone, p_id (fk)
13. s_id, s_type, p_id (fk), pl_id (fk)
14. r_id, distance, pl_id (fk)

Table Creation

1. CREATE TABLE manager (mgr_id NUMBER(3) CONSTRAINT pk_manager PRIMARY KEY, mgr_name VARCHAR2(30) NOT NULL, mgr_sal FLOAT NOT NULL CONSTRAINT manager_sal check(mgr_sal between 50000 AND 100000));
2. CREATE TABLE manager_contact(mgr_phone NUMBER(11) CONSTRAINT pk_mgr_contact PRIMARY KEY, mgr_id NUMBER(3) NOT NULL, CONSTRAINT fk_mgr_contact FOREIGN KEY(mgr_id) REFERENCES manager(mgr_id));
3. CREATE TABLE agent(a_id NUMBER(3) CONSTRAINT pk_agent PRIMARY KEY, a_name VARCHAR2(30) NOT NULL, a_sal FLOAT NOT NULL CONSTRAINT agent_sal check(a_sal BETWEEN 20000 AND 50000), mgr_id NUMBER(3) NOT NULL, CONSTRAINT fk_agent FOREIGN KEY(mgr_id) REFERENCES manager(mgr_id));
4. CREATE TABLE agent_contact(a_phone NUMBER(11) CONSTRAINT pk_agent_contact PRIMARY KEY, a_id NUMBER(3) NOT NULL, CONSTRAINT fk_agent_contact FOREIGN KEY(a_id) REFERENCES agent(a_id));
5. CREATE TABLE address(cc_id NUMBER(3) CONSTRAINT pk_address PRIMARY KEY, city VARCHAR2(30) NOT NULL, country VARCHAR2(30) NOT NULL);

6. CREATE TABLE customer(c_id NUMBER(10) CONSTRAINT pk_customer PRIMARY KEY, c_name VARCHAR2(30) NOT NULL, c_age NUMBER(3) NOT NULL CONSTRAINT c_age CHECK(c_age>0), c_gender VARCHAR2(6) NOT NULL CONSTRAINT c_gender CHECK (c_gender IN ('MALE', 'FEMALE', 'OTHERS')), pay_history NUMBER(10), cc_id NUMBER(3) NOT NULL, CONSTRAINT fk_address_customer FOREIGN KEY (cc_id) REFERENCES address(cc_id));
7. CREATE TABLE customer_contact(c_phone NUMBER(11) CONSTRAINT pk_c_contact PRIMARY KEY, c_id NUMBER(10) NOT NULL, CONSTRAINT fk_c_contact FOREIGN KEY(c_id) REFERENCES customer(c_id));
8. CREATE TABLE ticket(t_id NUMBER(10) CONSTRAINT pk_ticket PRIMARY KEY, amount NUMBER(2) NOT NULL, fare FLOAT NOT NULL, departure VARCHAR2(100) NOT NULL, destination VARCHAR2(100) NOT NULL, p_date DATE NOT NULL, d_date DATE NOT NULL, baggage NUMBER(2) NOT NULL, a_id NUMBER(3) NOT NULL, c_id NUMBER(10) NOT NULL, CONSTRAINT fk_agent_ticket FOREIGN KEY (a_id) REFERENCES agent (a_id),CONSTRAINT fk_customer_ticket FOREIGN KEY (c_id) REFERENCES customer (c_id));
9. CREATE TABLE payment(pay_id NUMBER(10) CONSTRAINT pk_payment PRIMARY KEY, t_fare FLOAT NOT NULL, c_id NUMBER(10) NOT NULL, t_id NUMBER(10) NOT NULL, CONSTRAINT fk_customer_payment FOREIGN KEY (c_id) REFERENCES customer (c_id), CONSTRAINT fk_ticket_payment FOREIGN KEY (t_id) REFERENCES ticket(t_id));
10. CREATE TABLE plane(pl_id NUMBER(3) CONSTRAINT pk_plane PRIMARY KEY, t_no NUMBER(3) NOT NULL, takeoff_airport VARCHAR2(100) NOT NULL, takeoff_time DATE NOT NULL, d_airport VARCHAR2(100) NOT NULL, capacity NUMBER(3) NOT NULL);
11. CREATE TABLE passenger(p_id NUMBER(10) CONSTRAINT pk_passenger PRIMARY KEY, p_name VARCHAR2(30) NOT NULL, p_gender VARCHAR2(6) NOT NULL CONSTRAINT p_gender CHECK (p_gender IN ('MALE', 'FEMALE', 'OTHERS')), p_age NUMBER(3) NOT NULL CONSTRAINT p_age CHECK(p_age>0), c_id NUMBER(10) NOT NULL, pl_id NUMBER(3) NOT NULL, cc_id NUMBER(3) NOT NULL, CONSTRAINT fk_customer_passenger FOREIGN KEY (c_id) REFERENCES customer(c_id), CONSTRAINT fk_plane_passenger FOREIGN KEY (pl_id) REFERENCES plane (pl_id), CONSTRAINT fk_address_passenger FOREIGN KEY (cc_id) REFERENCES address(cc_id));
12. CREATE TABLE passenger_contact(p_phone NUMBER(11) CONSTRAINT pk_p_contact PRIMARY KEY, p_id NUMBER(10) NOT NULL, CONSTRAINT fk_p_contact FOREIGN KEY(p_id) REFERENCES passenger(p_id));

13. CREATE TABLE seat(s_id NUMBER(3) CONSTRAINT pk_seat PRIMARY KEY, s_type VARCHAR2(20) NOT NULL, p_id NUMBER(10) NOT NULL, pl_id NUMBER(3) NOT NULL, CONSTRAINT fk_passenger_seat FOREIGN KEY (p_id) REFERENCES passenger(p_id), CONSTRAINT fk_plane_seat FOREIGN KEY (pl_id) REFERENCES plane (pl_id));
14. CREATE TABLE route(r_id NUMBER(3) CONSTRAINT pk_route PRIMARY KEY, distance VARCHAR2(10) NOT NULL, pl_id NUMBER(3) NOT NULL, CONSTRAINT fk_plane_route FOREIGN KEY (pl_id) REFERENCES plane (pl_id));

Table Creation (Screenshot)

1. Manager

The screenshot shows the Oracle Database Express Edition interface. The SQL Commands window contains the following SQL statement:

```
CREATE TABLE manager (mgr_id NUMBER(3) CONSTRAINT pk_manager PRIMARY KEY, mgr_name VARCHAR2(30) NOT NULL, mgr_sal FLOAT NOT NULL
CONSTRAINT manager_sal check(mgr_sal between 50000 AND 100000));
```

Below the SQL window, the 'Describe' tab is selected, showing the table structure for the MANAGER table:

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MANAGER	MGR_ID	Number	-	3	0	1	-	-	-
	MGR_NAME	Varchar2	30	-	-	-	-	-	-
	MGR_SAL	Float	22	126	-	-	-	-	-

The status bar at the bottom indicates 'Application Express 2.1.0.00.39' and the time '10:30 PM 5/14/2023'.

2. Manager Contact

The screenshot shows the Oracle Database Express Edition interface. The SQL Commands window contains the following SQL statement:

```
CREATE TABLE manager_contact (mgr_phone NUMBER(11) CONSTRAINT pk_mgr_contact PRIMARY KEY, mgr_id NUMBER(3) NOT NULL, CONSTRAINT
fk_mgr_contact FOREIGN KEY (mgr_id) REFERENCES manager(mgr_id));
```

Below the SQL window, the 'Describe' tab is selected, showing the table structure for the MANAGER_CONTACT table:

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MANAGER_CONTACT	MGR_PHONE	Number	-	11	0	1	-	-	-
	MGR_ID	Number	-	3	0	-	-	-	-

The status bar at the bottom indicates 'Application Express 2.1.0.00.39' and the time '10:30 AM 5/15/2023'.

3. Agent

Oracle Database Express Edition

User: URAN

Home > SQL > SQL Commands

Autocommit Display 10

Save Run

```
CREATE TABLE agent(a_id NUMBER(3) CONSTRAINT pk_agent PRIMARY KEY, a_name VARCHAR2(30) NOT NULL, a_sal FLOAT NOT NULL CONSTRAINT agent_sal check(a_sal BETWEEN 20000 AND 50000), mgr_id NUMBER(3) NOT NULL, CONSTRAINT fk_agent FOREIGN KEY(mgr_id) REFERENCES manager(mgr_id));
```

desc agent;

Results Explain Describe Saved SQL History

Object Type TABLE Object AGENT

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
AGENT	A_ID	Number	-	3	0	1	-	-	-
	A_NAME	Varchar2	30	-	-	-	-	-	-
	A_SAL	Float	22	126	-	-	-	-	-
	MGR_ID	Number	-	3	0	-	-	-	-

1-4

4. Agent Contact

Oracle Database Express Edition

User: URAN

Home > SQL > SQL Commands

Autocommit Display 10

Save Run

```
CREATE TABLE agent_contact(a_phone NUMBER(11) CONSTRAINT pk_agent_contact PRIMARY KEY, a_id NUMBER(3) NOT NULL, CONSTRAINT fk_agent_contact FOREIGN KEY(a_id) REFERENCES agent(a_id));
```

desc agent_contact;

Results Explain Describe Saved SQL History

Object Type TABLE Object AGENT_CONTACT

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
AGENT_CONTACT	A_PHONE	Number	-	11	0	1	-	-	-
	A_ID	Number	-	3	0	-	-	-	-

1-2

5. Address

Oracle Database Express Edition

User: URAN

Home > SQL > SQL Commands

Autocommit Display 10

Save Run

```
CREATE TABLE address(cc_id NUMBER(3) CONSTRAINT pk_address PRIMARY KEY, city VARCHAR2(30) NOT NULL, country VARCHAR2(30) NOT NULL);
```

desc address;

Results Explain Describe Saved SQL History

Object Type TABLE Object ADDRESS

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
ADDRESS	CC_ID	Number	-	3	0	1	-	-	-
	CITY	Varchar2	30	-	-	-	-	-	-
	COUNTRY	Varchar2	30	-	-	-	-	-	-

1-3

6. Customer

SQL Commands

127.0.0.1:8080/uran/.../45001003/36057379905211/NO1003

Autocommit Display 10 Save Run

```
CREATE TABLE customer(c_id NUMBER(10) CONSTRAINT pk_customer PRIMARY KEY, c_name VARCHAR2(30) NOT NULL, c_age NUMBER(3) NOT NULL
CONSTRAINT c_age CHECK(c_age>0), c_gender VARCHAR2(6) NOT NULL CONSTRAINT c_gender CHECK (c_gender IN ('MALE', 'FEMALE', 'OTHERS')),
pay_history NUMBER(10), cc_id NUMBER(3) NOT NULL, CONSTRAINT fk_address_customer FOREIGN KEY (cc_id) REFERENCES address(cc_id));
```

desc customer;

Results Explain Describe Saved SQL History

Object Type TABLE Object CUSTOMER

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
CUSTOMER	C_ID	Number	-	10	0	1	-	-	-
	C_NAME	Varchar2	30	-	-	-	-	-	-
	C_AGE	Number	-	3	0	-	-	-	-
	C_GENDER	Varchar2	6	-	-	-	-	-	-
	PAY_HISTORY	Number	-	10	0	-	✓	-	-
	CC_ID	Number	-	3	0	-	-	-	-

1 - 6

Language: en-us

Application Express 2.1.0.00.39

Copyright © 1999, 2006, Oracle. All rights reserved.

7. Customer Contact

SQL Commands

127.0.0.1:8080/uran/.../45001003/7011040240381542/NO1003

User: URAN

Home > SQL > SQL Commands

Autocommit Display 10 Save Run

```
CREATE TABLE customer_contact(c_phone NUMBER(11) CONSTRAINT pk_c_contact PRIMARY KEY, c_id NUMBER(10) NOT NULL, CONSTRAINT fk_c_contact
FOREIGN KEY(c_id) REFERENCES customer(c_id));
```

desc customer_contact;

Results Explain Describe Saved SQL History

Object Type TABLE Object CUSTOMER_CONTACT

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
CUSTOMER_CONTACT	C_PHONE	Number	-	11	0	1	-	-	-
	C_ID	Number	-	10	0	-	-	-	-

1 - 2

Language: en-us

Application Express 2.1.0.00.39

Copyright © 1999, 2006, Oracle. All rights reserved.

8. Ticket

SQL Commands

127.0.0.1:8080/uran/.../45001003/3402751566177919/NO1003

Autocommit Display 10 Save Run

```
CREATE TABLE ticket(t_id NUMBER(10) CONSTRAINT pk_ticket PRIMARY KEY, amount NUMBER(2) NOT NULL, fare FLOAT NOT NULL, departure VARCHAR2(100) NOT NULL, destination
VARCHAR2(100) NOT NULL, p_date DATE NOT NULL, d_date DATE NOT NULL, baggage NUMBER(2) NOT NULL, a_id NUMBER(3) NOT NULL, c_id NUMBER(10) NOT NULL, CONSTRAINT
fk_agent_ticket FOREIGN KEY (a_id) REFERENCES agent (a_id),CONSTRAINT fk_customer_ticket FOREIGN KEY (c_id) REFERENCES customer (c_id));
```

desc ticket;

Results Explain Describe Saved SQL History

Object Type TABLE Object TICKET

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
TICKET	T_ID	Number	-	10	0	1	-	-	-
	AMOUNT	Number	-	2	0	-	-	-	-
	FARE	Float	22	126	-	-	-	-	-
	DEPARTURE	Varchar2	100	-	-	-	-	-	-
	DESTINATION	Varchar2	100	-	-	-	-	-	-
	P_DATE	Date	7	-	-	-	-	-	-
	D_DATE	Date	7	-	-	-	-	-	-
	BAGGAGE	Number	-	2	0	-	-	-	-
	A_ID	Number	-	3	0	-	-	-	-
	C_ID	Number	-	10	0	-	-	-	-

1 - 10

Language: en-us

Application Express 2.1.0.00.39

Copyright © 1999, 2006, Oracle. All rights reserved.

9. Payment

Oracle Database Express Edition

User: URAN

Home > SQL > SQL Commands

Autocommit: Display 10

Save Run

```
CREATE TABLE payment(pay_id NUMBER(10) CONSTRAINT pk_payment PRIMARY KEY, t_fare FLOAT NOT NULL, c_id NUMBER(10) NOT NULL, t_id NUMBER(10) NOT NULL, CONSTRAINT fk_customer_payment FOREIGN KEY (c_id) REFERENCES customer (c_id), CONSTRAINT fk_ticket_payment FOREIGN KEY (t_id) REFERENCES ticket(t_id));
```

desc payment;

Results Explain Describe Saved SQL History

Object Type TABLE Object PAYMENT

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PAYMENT	PAY_ID	Number	-	10	0	1	-	-	-
	T_FARE	Float	22	126	-	-	-	-	-
	C_ID	Number	-	10	0	-	-	-	-
	T_ID	Number	-	10	0	-	-	-	-

1 - 4

10. Plane

Oracle Database Express Edition

User: URAN

Home > SQL > SQL Commands

Autocommit: Display 10

Save Run

```
CREATE TABLE plane(pl_id NUMBER(3) CONSTRAINT pk_plane PRIMARY KEY, t_no NUMBER(3) NOT NULL, takeoff_airport VARCHAR2(100) NOT NULL, takeoff_time DATE NOT NULL, d_airport VARCHAR2(100) NOT NULL, capacity NUMBER(3) NOT NULL);
```

desc plane;

Results Explain Describe Saved SQL History

Object Type TABLE Object PLANE

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PLANE	PL_ID	Number	-	3	0	1	-	-	-
	T_NO	Number	-	3	0	-	-	-	-
	TAKEOFF_AIRPORT	Varchar2	100	-	-	-	-	-	-
	TAKEOFF_TIME	Date	7	-	-	-	-	-	-
	D_AIRPORT	Varchar2	100	-	-	-	-	-	-
	CAPACITY	Number	-	3	0	-	-	-	-

1 - 6

11. Passenger

Oracle Database Express Edition

User: URAN

Home > SQL > SQL Commands

Autocommit: Display 10

Save Run

```
CREATE TABLE passenger(p_id NUMBER(10) CONSTRAINT pk_passenger PRIMARY KEY, p_name VARCHAR2(30) NOT NULL, p_gender VARCHAR2(6) NOT NULL CONSTRAINT p_gender CHECK (p_gender IN ('MALE', 'FEMALE', 'OTHERS')), p_age NUMBER(3) NOT NULL CONSTRAINT p_age CHECK (p_age > 0), c_id NUMBER(10) NOT NULL, pl_id NUMBER(3) NOT NULL, cc_id NUMBER(3) NOT NULL, CONSTRAINT fk_customer_passenger FOREIGN KEY (c_id) REFERENCES customer(c_id), CONSTRAINT fk_plane_passenger FOREIGN KEY (pl_id) REFERENCES plane (pl_id), CONSTRAINT fk_address_passenger FOREIGN KEY (cc_id) REFERENCES address(cc_id));
```

desc passenger;

Results Explain Describe Saved SQL History

Object Type TABLE Object PASSENGER

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PASSENGER	P_ID	Number	-	10	0	1	-	-	-
	P_NAME	Varchar2	30	-	-	-	-	-	-
	P_GENDER	Varchar2	6	-	-	-	-	-	-
	P_AGE	Number	-	3	0	-	-	-	-
	C_ID	Number	-	10	0	-	-	-	-
	PL_ID	Number	-	3	0	-	-	-	-
	CC_ID	Number	-	3	0	-	-	-	-

1 - 7

12. Passenger Contact

Oracle Database Express Edition interface showing the creation and description of the **PASSENGER_CONTACT** table.

SQL Commands:

```
CREATE TABLE passenger_contact(p_phone NUMBER(11) CONSTRAINT pk_p_contact PRIMARY KEY, p_id NUMBER(10) NOT NULL, CONSTRAINT fk_p_contact FOREIGN KEY(p_id) REFERENCES passenger(p_id));
```

Describe Table:

```
desc passenger_contact;
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PASSENGER_CONTACT	P_PHONE	Number	11		0	1	-	-	-
	P_ID	Number	10		0	-	-	-	-

13. Seat

Oracle Database Express Edition interface showing the creation and description of the **SEAT** table.

SQL Commands:

```
CREATE TABLE seat(s_id NUMBER(3) CONSTRAINT pk_seat PRIMARY KEY, s_type VARCHAR2(20) NOT NULL, p_id NUMBER(10) NOT NULL, p1_id NUMBER(3) NOT NULL, CONSTRAINT fk_passenger_seat FOREIGN KEY (p_id) REFERENCES passenger(p_id), CONSTRAINT fk_plane_seat FOREIGN KEY (p1_id) REFERENCES plane (p1_id));
```

Describe Table:

```
desc seat;
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
SEAT	S_ID	Number	3		0	1	-	-	-
	S_TYPE	Varchar2	20		-	-	-	-	-
	P_ID	Number	10		0	-	-	-	-
	P1_ID	Number	3		0	-	-	-	-

14. Route

Oracle Database Express Edition interface showing the creation and description of the **ROUTE** table.

SQL Commands:

```
CREATE TABLE route(r_id NUMBER(3) CONSTRAINT pk_route PRIMARY KEY, distance VARCHAR2(10) NOT NULL, p1_id NUMBER(3) NOT NULL, CONSTRAINT fk_plane_route FOREIGN KEY (p1_id) REFERENCES plane (p1_id));
```

Describe Table:

```
desc route;
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
ROUTE	R_ID	Number	3		0	1	-	-	-
	DISTANCE	Varchar2	10		-	-	-	-	-
	P1_ID	Number	3		0	-	-	-	-

Data Insertion:

1. Manager

- a) INSERT INTO manager(mgr_id, mgr_name, mgr_sal) VALUES(100, 'Kawser', 99999);
- b) INSERT INTO manager(mgr_id, mgr_name, mgr_sal) VALUES(200, 'Irom', 89999);
- c) INSERT INTO manager(mgr_id, mgr_name, mgr_sal) VALUES(300, 'Rushee', 60000);

2. Manager Contact

- a) INSERT INTO manager_contact(mgr_phone, mgr_id) VALUES(01765238741, 100);
- b) INSERT INTO manager_contact(mgr_phone, mgr_id) VALUES(01768738741, 200);
- c) INSERT INTO manager_contact(mgr_phone, mgr_id) VALUES(01763438741, 300);

3. Agent

- a) INSERT INTO agent(a_id, a_name, a_sal, mgr_id) VALUES(459, 'Niloy', 30000, 100);
- b) INSERT INTO agent(a_id, a_name, a_sal, mgr_id) VALUES(588, 'Azminur', 45000, 300);
- c) INSERT INTO agent(a_id, a_name, a_sal, mgr_id) VALUES(625, 'Sakib', 40000, 200);
- d) INSERT INTO agent(a_id, a_name, a_sal, mgr_id) VALUES(615, 'Saikot', 35000, 200);

4. Agent Contact

- a) INSERT INTO agent_contact(a_phone, a_id) VALUES(01763438778, 459);
- b) INSERT INTO agent_contact(a_phone, a_id) VALUES(01763438711, 588);
- c) INSERT INTO agent_contact(a_phone, a_id) VALUES(01763438767, 625);
- d) INSERT INTO agent_contact(a_phone, a_id) VALUES(01763438744, 615);

5. Address

- a) INSERT INTO address(cc_id, city, country) VALUES(10, 'Dhaka', 'Bangladesh');
- b) INSERT INTO address(cc_id, city, country) VALUES(20, 'Chittagong', 'Bangladesh');
- c) INSERT INTO address(cc_id, city, country) VALUES(30, 'Rajshahi', 'Bangladesh');
- d) INSERT INTO address(cc_id, city, country) VALUES(40, 'Khulna', 'Bangladesh');
- e) INSERT INTO address(cc_id, city, country) VALUES(50, 'Barishal', 'Bangladesh');
- f) INSERT INTO address(cc_id, city, country) VALUES(60, 'Sylhet', 'Bangladesh');
- g) INSERT INTO address(cc_id, city, country) VALUES(70, 'Rangpur', 'Bangladesh');
- h) INSERT INTO address(cc_id, city, country) VALUES(80, 'Mymensingh', 'Bangladesh');

6. Customer

- a) INSERT INTO customer(c_id, c_name, c_age, c_gender, pay_history, cc_id) VALUES(22466251, 'Shaon', 22, 'MALE', 9817, 30);
- b) INSERT INTO customer(c_id, c_name, c_age, c_gender, pay_history, cc_id) VALUES(22465881, 'Rahman', 22, 'MALE', 6591, 40);
- c) INSERT INTO customer(c_id, c_name, c_age, c_gender, pay_history, cc_id) VALUES(22464591, 'Nafiur', 69, 'MALE', 23221, 80);
- d) INSERT INTO customer(c_id, c_name, c_age, c_gender, pay_history, cc_id) VALUES(22464592, 'Kundu', 15, 'MALE', 21231, 30);

7. Customer Contact

- a) INSERT INTO customer_contact(c_phone, c_id) VALUES(01954879621, 22465881);
- b) INSERT INTO customer_contact(c_phone, c_id) VALUES(01935489647, 22466251);
- c) INSERT INTO customer_contact(c_phone, c_id) VALUES(01821365482, 22464591);
- d) INSERT INTO customer_contact(c_phone, c_id) VALUES(01132654791, 22464592);

8. Ticket

- a) INSERT INTO ticket(t_id, amount, fare, departure, destination, p_date, d_date, baggage, a_id, c_id) VALUES(45698745, 1, 4699, 'DHK', 'JES', TO_DATE('08-05-2023', 'DD-MM-YYYY'), TO_DATE('15-05-2023', 'DD-MM-YYYY'), 20, 459, 22466251);
- b) INSERT INTO ticket(t_id, amount, fare, departure, destination, p_date, d_date, baggage, a_id, c_id) VALUES(45698565, 2, 9398, 'DHK', 'JES', TO_DATE('09-05-2023', 'DD-MM-YYYY'), TO_DATE('15-05-2023', 'DD-MM-YYYY'), 30, 588, 22465881);
- c) INSERT INTO ticket(t_id, amount, fare, departure, destination, p_date, d_date, baggage, a_id, c_id) VALUES(45698234, 2, 9398, 'DHK', 'JES', TO_DATE('11-05-2023', 'DD-MM-YYYY'), TO_DATE('15-05-2023', 'DD-MM-YYYY'), 50, 615, 22464591);

9. Payment

- a) INSERT INTO payment(pay_id, t_fare, c_id, t_id) VALUES(1578811645, 4699, 22466251, 45698745);
- b) INSERT INTO payment(pay_id, t_fare, c_id, t_id) VALUES(1578845682, 9398, 22465881, 45698565);
- c) INSERT INTO payment(pay_id, t_fare, c_id, t_id) VALUES(1574568214, 9398, 22464591, 45698234);

10. Plane

- a) INSERT INTO plane(pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity) VALUES(777, 13, 'DHK', TO_DATE('15-05-2023 09:30:00', 'DD-MM-YYYY HH24:MI:SS'), 'JES', 150);

- b) INSERT INTO plane(pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity) VALUES(787, 10, 'DHK', TO_DATE('17-05-2023 12:30:00', 'DD-MM-YYYY HH24:MI:SS'), 'CHI', 150);
- c) INSERT INTO plane(pl_id, t_no, takeoff_airport, takeoff_time, d_airport, capacity) VALUES(666, 02, 'DHK', TO_DATE('19-05-2023 16:00:00', 'DD-MM-YYYY HH24:MI:SS'), 'RAJ', 150);

11. Passenger

- a) INSERT INTO passenger(p_id, p_name, p_gender, p_age, c_id, pl_id, cc_id) VALUES(22466251, 'Shaon', 'MALE', 22, 22466251, 777, 30);
- b) INSERT INTO passenger(p_id, p_name, p_gender, p_age, c_id, pl_id, cc_id) VALUES(22465881, 'Rahman', 'MALE', 22, 22465881, 777, 40);
- c) INSERT INTO passenger(p_id, p_name, p_gender, p_age, c_id, pl_id, cc_id) VALUES(22464591, 'Nafiur', 'MALE', 69, 22464591, 777, 80);
- d) INSERT INTO passenger(p_id, p_name, p_gender, p_age, c_id, pl_id, cc_id) VALUES(16989898, 'Neela', 'FEMALE', 18, 22464591, 777, 50);
- e) INSERT INTO passenger(p_id, p_name, p_gender, p_age, c_id, pl_id, cc_id) VALUES(51897848, 'Saikot', 'MALE', 25, 22465881, 777, 70);

12. Passenger Contact

- a) INSERT INTO passenger_contact(p_phone, p_id) VALUES(01935489647, 22466251);
- b) INSERT INTO passenger_contact(p_phone, p_id) VALUES(01954879621, 22465881);
- c) INSERT INTO passenger_contact(p_phone, p_id) VALUES(01821365482, 22464591);
- d) INSERT INTO passenger_contact(p_phone, p_id) VALUES(01334567890, 16989898);
- e) INSERT INTO passenger_contact(p_phone, p_id) VALUES(01434567890, 51897848);

13. Seat

- a) INSERT INTO seat(s_id, s_type, p_id, pl_id) VALUES(101, 'Standard', 22466251, 777);
- b) INSERT INTO seat(s_id, s_type, p_id, pl_id) VALUES(102, 'Business', 22465881, 777);
- c) INSERT INTO seat(s_id, s_type, p_id, pl_id) VALUES(503, 'Economy', 22464591, 777);
- d) INSERT INTO seat(s_id, s_type, p_id, pl_id) VALUES(504, 'Economy', 16989898, 777);
- e) INSERT INTO seat(s_id, s_type, p_id, pl_id) VALUES(103, 'Business', 51897848, 777);

14. Route

- a) INSERT INTO route(r_id, distance, pl_id) VALUES(100, '144 KM', 777);
- b) INSERT INTO route(r_id, distance, pl_id) VALUES(200, '227 KM', 787);
- c) INSERT INTO route(r_id, distance, pl_id) VALUES(300, '192 KM', 666);

Joining

1. Equijoin

Question: "Find customers who have passengers with the same gender as themselves."

Query:

```
SELECT c.c_name, c.c_gender, p.p_name, p.p_gender FROM customer c, passenger p
WHERE c.c_id = p.c_id AND c.c_gender = p.p_gender;
```

Screenshot:

Oracle Database Express Edition interface showing the execution of the following SQL query:

```
SELECT c.c_name, c.c_gender, p.p_name, p.p_gender
FROM customer c, passenger p
WHERE c.c_id = p.c_id AND c.c_gender = p.p_gender;
```

The results table displays the following data:

C_NAME	C_GENDER	P_NAME	P_GENDER
Shaon	MALE	Shaon	MALE
Rahman	MALE	Rahman	MALE
Nafiur	MALE	Nafiur	MALE
Rahman	MALE	Saikot	MALE

4 rows returned in 0.00 seconds

2. Outer Join

Left:

Question: List all customers and their corresponding tickets, including those customers who do not have any tickets.

Query:

```
SELECT c.c_name, t.t_id FROM customer c, ticket t WHERE c.c_id = t.c_id (+);
```

Screenshot:

Oracle Database Express Edition interface showing the execution of the following SQL query:

```
SELECT c.c_name, t.t_id FROM customer c, ticket t WHERE c.c_id = t.c_id (+);
```

The results table displays the following data:

C_NAME	T_ID
Nafiur	45698234
Kundu	-
Rahman	45698565
Shaon	45698745

4 rows returned in 0.00 seconds

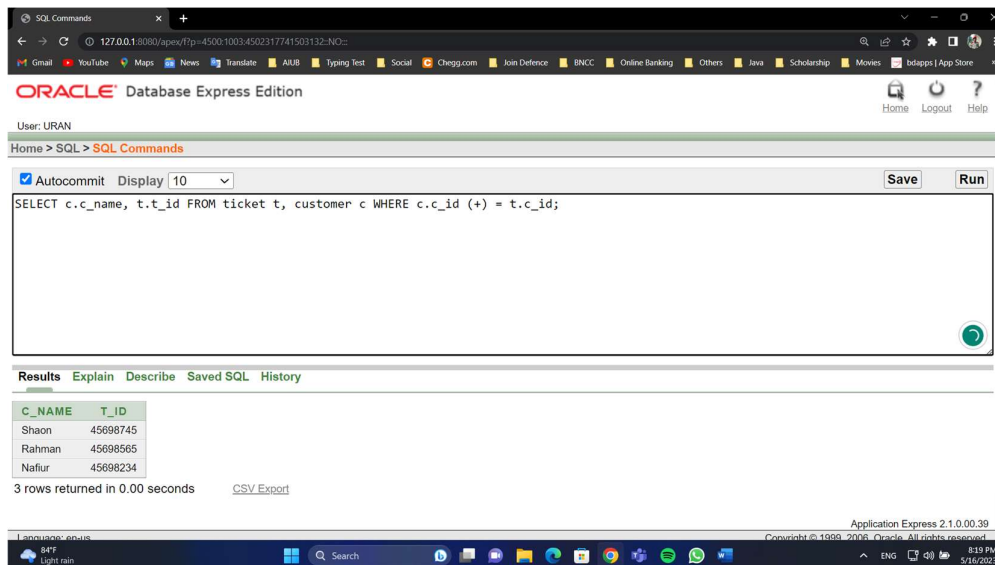
Right:

Question: List all tickets and their corresponding customers, including those tickets that are not associated with any customer.

Query:

SELECT c.c_name, t.t_id FROM ticket t, customer c WHERE c.c_id (+) = t.c_id;

Screenshot:



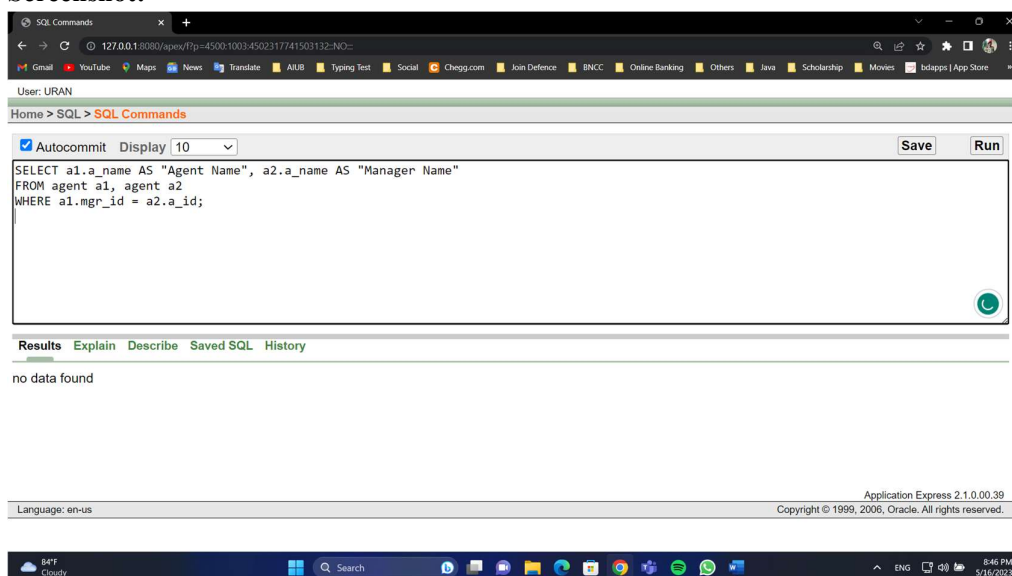
3. Self-Join

Question: Retrieve the agent names and their corresponding manager names from the agent table where the agent is managed by another agent.

Query:

SELECT a1.a_name AS "Agent Name", a2.a_name AS "Manager Name"
FROM agent a1, agent a2
WHERE a1.mgr_id = a2.a_id;

Screenshot:

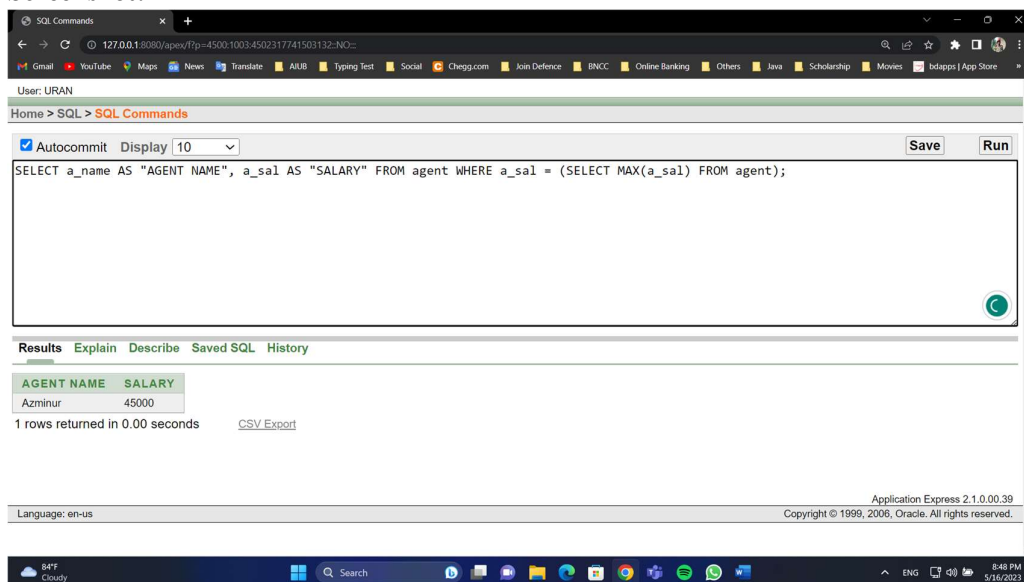


Subquery

1. **Question:** "Retrieve the agent name & salary who has the highest salary among all agents."

Query: SELECT a_name AS "AGENT NAME", a_sal AS "SALARY" FROM agent WHERE a_sal = (SELECT MAX(a_sal) FROM agent);

Screenshot:



The screenshot shows a web browser window with the URL `127.0.0.1:8080/apex/f?p=4500:1003:4502317741503132::NO::`. The page title is "SQL Commands". The user is "URAN". The page shows the query: `SELECT a_name AS "AGENT NAME", a_sal AS "SALARY" FROM agent WHERE a_sal = (SELECT MAX(a_sal) FROM agent);`. The results are displayed in a table with two columns: "AGENT NAME" and "SALARY". The result is:

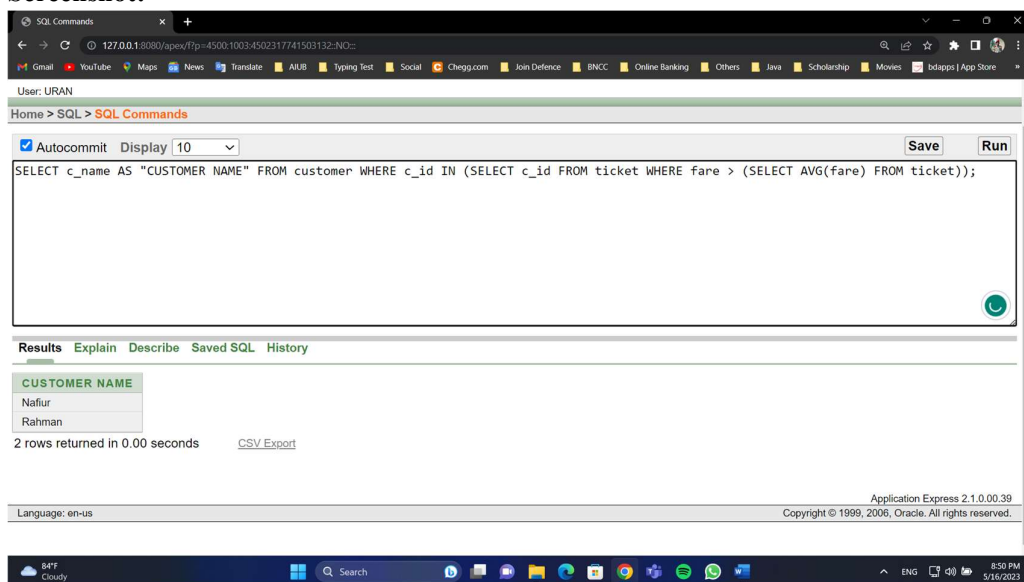
AGENT NAME	SALARY
Azminur	45000

. Below the table, it says "1 rows returned in 0.00 seconds" and "CSV Export". The footer of the application says "Application Express 2.1.0.00.39" and "Copyright © 1999, 2006, Oracle. All rights reserved."

2. **Question:** "Find the customer's name who have purchased tickets with a fare greater than the average fare across all tickets."

Query: SELECT c_name AS "CUSTOMER NAME" FROM customer WHERE c_id IN (SELECT c_id FROM ticket WHERE fare > (SELECT AVG(fare) FROM ticket));

Screenshot:



The screenshot shows a web browser window with the URL `127.0.0.1:8080/apex/f?p=4500:1003:4502317741503132::NO::`. The page title is "SQL Commands". The user is "URAN". The page shows the query: `SELECT c_name AS "CUSTOMER NAME" FROM customer WHERE c_id IN (SELECT c_id FROM ticket WHERE fare > (SELECT AVG(fare) FROM ticket));`. The results are displayed in a table with one column: "CUSTOMER NAME". The results are:

CUSTOMER NAME
Nafur
Rahman

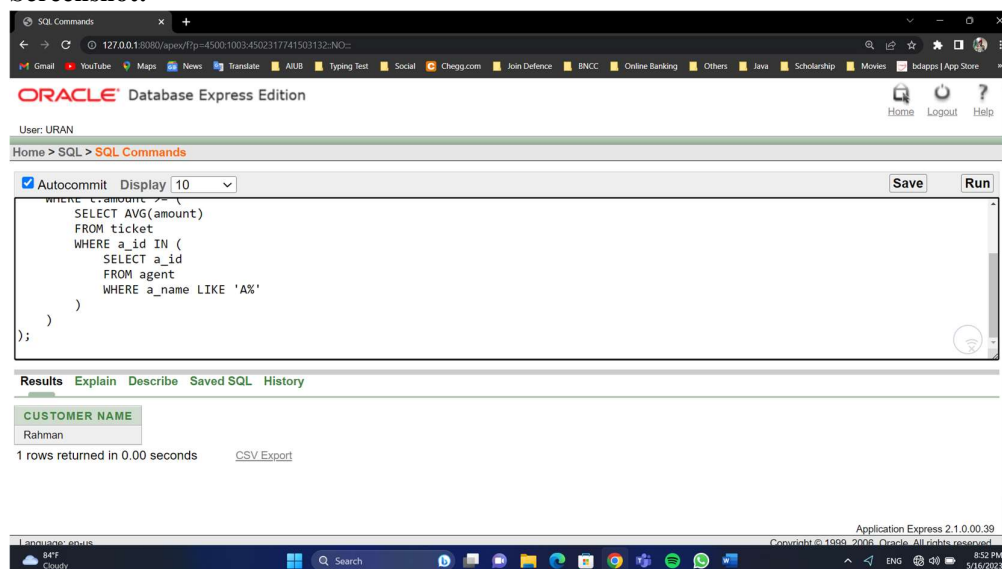
. Below the table, it says "2 rows returned in 0.00 seconds" and "CSV Export". The footer of the application says "Application Express 2.1.0.00.39" and "Copyright © 1999, 2006, Oracle. All rights reserved."

3. **Question:** Find the names of customers whose names end with the letter 'n' and who have booked tickets with an amount greater than or equal to the average amount of tickets booked by agents whose names start with 'A'.

Query:

```
SELECT c.c_name AS "CUSTOMER NAME"
FROM customer c
WHERE c.c_name LIKE '%n' AND c.c_id IN (
    SELECT t.c_id
    FROM ticket t
    WHERE t.amount >= (
        SELECT AVG(amount)
        FROM ticket
        WHERE a_id IN (
            SELECT a_id
            FROM agent
            WHERE a_name LIKE 'A%'
        )
    )
);
```

Screenshot:



View

1. Complex View

Question: Create a complex view that combines information from multiple tables to provide a comprehensive overview of customers and their corresponding ticket details.

Query:

```
CREATE VIEW customer_ticket_view AS
SELECT c.c_id, c.c_name, t.t_id, t.amount, t.fare, t.departure, t.destination
FROM customer c, ticket t
WHERE c.c_id = t.c_id;
```

Screenshot:

The screenshot shows the Oracle Database Express Edition interface. The SQL Commands window contains the following code:

```
CREATE VIEW customer_ticket_view AS
SELECT c.c_id, c.c_name, t.t_id, t.amount, t.fare, t.departure, t.destination
FROM customer c, ticket t
WHERE c.c_id = t.c_id;

select * from customer_ticket_view;
```

The Results window displays the following data:

C_ID	C_NAME	T_ID	AMOUNT	FARE	DEPARTURE	DESTINATION
22406251	Shaan	45608745	1	4699	DHK	JES
22405881	Rahman	45608565	2	9398	DHK	JES
22404591	Nafur	45608234	2	9398	DHK	JES

3 rows returned in 0.00 seconds

2. Simple View

Question: Create a simple view that displays the names and salaries of managers.

Query:

```
CREATE VIEW manager_salary_view AS
SELECT mgr_name, mgr_sal
FROM manager;
```

Screenshot:

The screenshot shows the Oracle Database Express Edition interface. The SQL Commands window contains the following code:

```
CREATE VIEW manager_salary_view AS
SELECT mgr_name, mgr_sal
FROM manager;

select * from manager_salary_view;
```

The Results window displays the following data:

MGR_NAME	MGR_SAL
Kawser	99999
Irom	89999
Rushee	60000

3 rows returned in 0.00 seconds

Add Constraint

Question: Add constraint in customer table payment history and it must be positive or 0 and not null.

Query:

```
ALTER TABLE customer
ADD CONSTRAINT check_pay_history_positive
CHECK (pay_history >= 0 AND pay_history IS NOT NULL);
```

Screenshot:

