

# **DELHI TECHNOLOGICAL UNIVERSITY**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



**CO 202: DATABASE MANAGEMENT SYSTEMS**

## **PRACTICAL FILE**

**Submitted To:**

**Ms. Indu Singh**

**Assistant Professor**

**Department of CSE, DTU**

**Submitted By:**

**Neeraj Sharma**

**2K19/CO/244**

**A4 Batch Group 1**

## **TABLE OF CONTENTS**

<b>S.NO</b>	<b>Title</b>	<b>Date</b>	<b>Page No.</b>
5	Case Study 1: Railway Management System	March 12, 2021	3 – 4
6	Creating Entity-Relationship Diagram, Entity Tables and Relationship Tables	Match 12, 2021	5-7
7.	Implementing DDL and DML commands.	March 22, 2021	8-17
8.	Performing Simple Queries.	March 29, 2021	17-20
9.	Implementation of Simple Joins	March 31, 2021	20-24

## **EXPERIMENT 5**

### **CASE STUDY: RAILWAY MANAGEMENT SYSTEM.**

#### **PROBLEM STATEMENT:**

- Railway Reservation System will maintain the information about Reservation like Passenger name, Id etc.
- Railway Reservation System will maintain the information about passenger like Personal details and Official detail.
- System will maintain the information about train like Train no., Train name.
- Railway Reservation System will maintain the information about the seat availability like Source station, Destination station, Date, Train no., Class, Arrival time.
- Railway Reservation System will maintain the information about the Account like, Date, PNR and Amount.
- Railway Reservation System will maintain the information about the Train for cancellation like PNR, No. of seat, Date.

#### **ASSUMPTIONS:**

- One Passenger can reserve N seats in one train.
- One Passenger can pay the Ticket Rate by only one account.
- One Passenger can cancel the N seats at one time. □ Passenger details can be Personal or Official.

#### **ER DIAGRAM DESCRIPTION:**

##### **➤ Identification of Entities:**

1. PASSENGER (Personal Id, Name, Age, Sex, Phone No, Address, Office Name, Office Address, Office Phone)
2. TRAIN (Train No., Train Name, Duration, First Station, Last Station)
3. RESERVATION (Reservation No., Passenger Id, Passenger Name, Train Number, Number of Seats, Reservation Date)
4. PASS\_ACCOUNT (PNR, Account Holder Name, Date of Booking, Amount)
5. CANCELLATION (Cancellation No., PNR, Train No., Train Name, Class, Number of Seats, Reservation Date)

6. SEAT AVAILABILITY (Train No., Total Seats, Reservation Date, Class, Quota, Available Seats)
7. TICKET (Train No., Ticket Price)

➤ **Identification of Relationships:**

1. RESERVATION: PASSENGER (N:1)
2. RESERVATION: TRAIN (N:M)
3. TRAIN: SEAT AVAILABILITY (N:M)
4. TRAIN: PASS\_ACCOUNT (N:1)

**CONCLUSION:**

This experiment helped us to analyse the problem of the case study and make certain assumptions that will be helpful in studying the problem. It also helped us identify the key entities and their respective attributes which can be used to create relations. It helps us to construct ER diagrams.

## **EXPERIMENT 6**

**AIM:** Creating Entity-Relationship Diagram, Entity Tables and Relationship Tables

### **THEORY**

Entity Relationship Diagrams are used to define the relationships between various entities. The Entities and Relationships had been identified in Experiment 5. As observed, there are seven key entities and four major relationships. They are listed below.

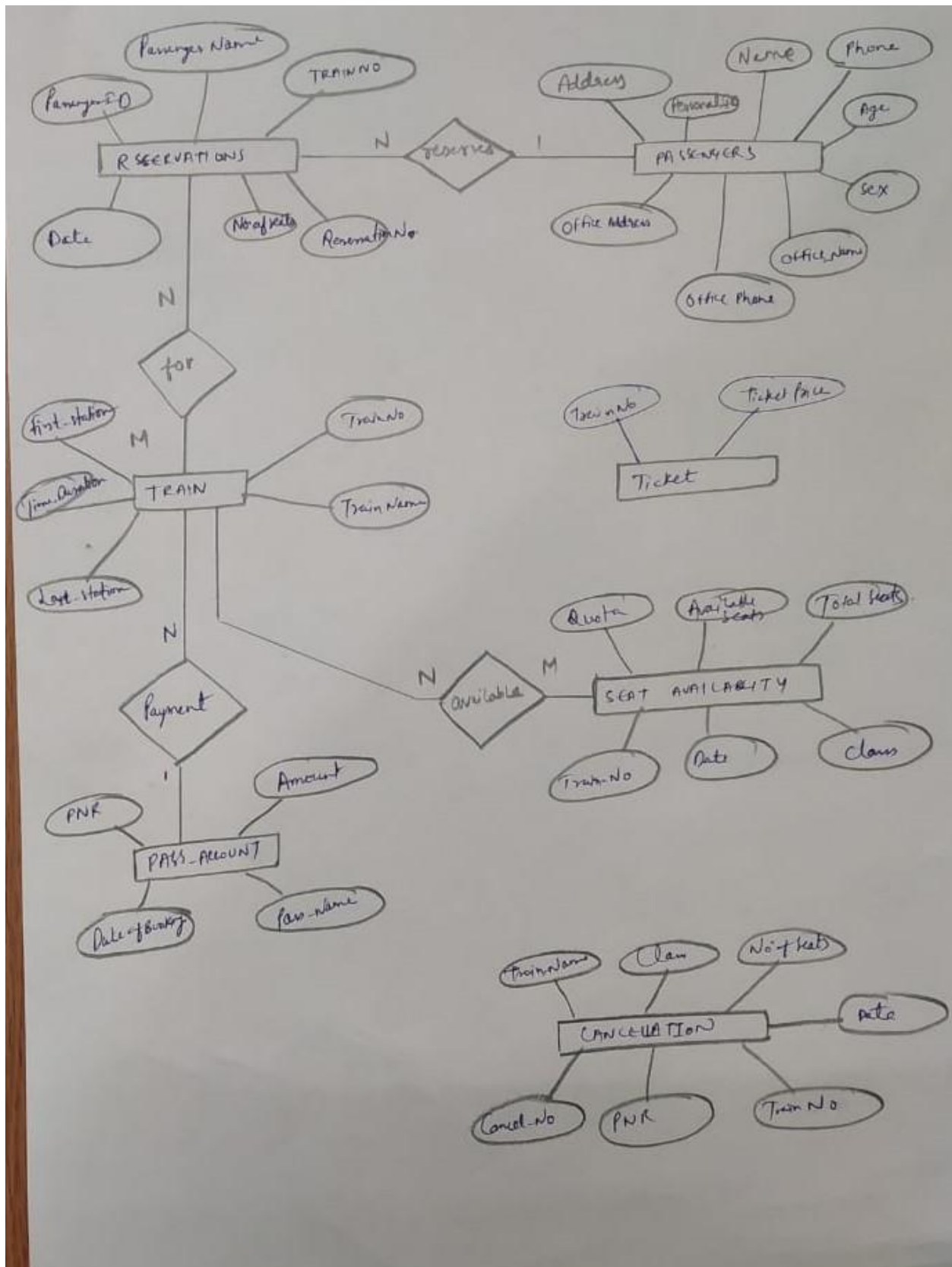
➤ **Identification of Entities:**

1. PASSENGER (Personal Id, Name, Age, Sex, Phone No, Address, Office Name, Office Address, Office Phone)
2. TRAIN (Train No., Train Name, Duration, First Station, Last Station)
3. RESERVATION (Reservation No., Passenger Id, Passenger Name, Train Number, Number of Seats, Reservation Date)
4. PASS\_ACCOUNT (PNR, Account Holder Name, Date of Booking, Amount)
5. CANCELLATION (Cancellation No., PNR, Train No., Train Name, Class, Number of Seats, Reservation Date)
6. SEAT AVAILABILITY (Train No., Total Seats, Reservation Date, Class, Quota, Available Seats)
7. TICKET (Train No., Ticket Price)

➤ **Identification of Relationships:**

1. RESERVATION: PASSENGER (N:1)
2. RESERVATION: TRAIN (N:M)
3. TRAIN: SEAT AVAILABILITY (N:M)
4. TRAIN: PASS\_ACCOUNT (N:1)

The relevant Entity Relationship Diagram has been shown below.



**CONCLUSION:**

This experiment helped us simplify the case study and identify the number of table that need to be made, the attribute that they should contain as well as visualize the relationships between them.

## **EXPERIMENT 7**

**AIM:** Implementing DDL and DML commands.

### **THEORY**

#### **DDL Commands:**

DDL stands for Database Definition Language. These types of SQL commands are used to define the database schema. It is used to create or modify database objects and deals with the description of the database schema. The most commonly used DDL commands are:

1. **CREATE TABLE:** This command is used to create a new relation in the database.

**Syntax:** CREATE TABLE table\_name

(Attribute1, Datatype,  
Attribute2, Datatype...)

2. **ALTER TABLE:** This command is used to add, delete or manipulate attributes in a relation.

**Syntax:** ALTER TABLE table\_name

ADD Attribute\_Name datatype;

3. **DROP TABLE:** This command is used to remove or delete a relation from a database

**Syntax:** DROP TABLE table\_name;

4. **TRUNCATE:** This command is used to erase complete data from an existing relation  
**Syntax:** TRUNCATE TABLE table\_name;



## Implementing DDL Commands:

We will be implementing the CREATE and ALTER Table commands for this case study. The results are shown below.

### CREATE:

#### 1. PASSENGERS TABLE:

Home > SQL > SQL Commands

☒ Autocommit Display 10 ▼

```
CREATE TABLE PASSENGERS
(
PERSONAL_ID INT NOT NULL PRIMARY KEY,
NAME VARCHAR(30) NOT NULL,
SEX VARCHAR(10) NOT NULL,
AGE INT NOT NULL CHECK(AGE<=110),
PHONE_NO VARCHAR2(10),
ADDRESS VARCHAR(50),
OFFICE_NAME VARCHAR(30),
OFFICE_PHONE VARCHAR(10),
OFFICE_ADDRESS VARCHAR(50)
);|
```

Results Explain Describe Saved SQL History

Table created.

0.02 seconds

## 2. RESERVATION TABLE:

Home > SQL > SQL Commands

☒ Autocommit Display 10 ▼

```
CREATE TABLE RESERVATION
(
  RESERVATION_NO INT NOT NULL PRIMARY KEY,
  PASSENGER_ID INT NOT NULL,
  PASSENGER_NAME VARCHAR(40) NOT NULL,
  TRAIN_NUM INT NOT NULL,
  NUMBER_OF_SEATS INT NOT NULL,
  RESERVATION_DATE DATE NOT NULL,
  FOREIGN KEY(TRAIN_NUM) REFERENCES TRAIN(TRAIN_NO)
);
```

[Results](#) [Explain](#) [Describe](#) [Saved SQL](#) [History](#)

Table created.

0.01 seconds

## 3. TRAIN TABLE:

Home > SQL > SQL Commands

☒ Autocommit Display 10 ▼

```
CREATE TABLE TRAIN
(
  TRAIN_NO INT NOT NULL PRIMARY KEY,
  TRAIN_NAME VARCHAR(50) NOT NULL,
  TIME_DURATION INT,
  FIRST_STATION VARCHAR(40),
  LAST_STATION VARCHAR(40)
);
```

Results Explain Describe Saved SQL History

Table created.

0.00 seconds

#### 4. TICKET TABLE:

☒ Autocommit Display 10 ▼

```
CREATE TABLE TICKET
(
  TRAIN_NUMBER INT NOT NULL,
  TICKET_PRICE INT NOT NULL,
  FOREIGN KEY(TRAIN_NUMBER) REFERENCES TRAIN(TRAIN_NO)
);
```

Results Explain Describe Saved SQL History

Table created.

0.00 seconds

#### 5. PASS ACCOUNT TABLE:

Home > SQL > SQL Commands

☒ Autocommit Display 10 ▼

```
CREATE TABLE PASS_ACCOUNT
(
  PNR INT NOT NULL PRIMARY KEY,
  PASS_NAME VARCHAR(50) NOT NULL,
  DATE_OF_BOOKING VARCHAR(10) NOT NULL,
  AMOUNT INT NOT NULL
);
```

Results Explain Describe Saved SQL History

Table created.

0.02 seconds

## 6. CANCELLATION TABLE:

Home > SQL > SQL Commands

☒ Autocommit Display 10 ▼

```
CREATE TABLE CANCELLATION
(
  PNR_NUM INT NOT NULL,
  TRAIN_NUM INT NOT NULL,
  TRAIN_NAME VARCHAR(50) NOT NULL,
  CLASS VARCHAR(15) NOT NULL,
  CANCEL_NO INT NOT NULL PRIMARY KEY,
  NUMBER_OF_SEATS INT NOT NULL,
  RES_DATE DATE NOT NULL,
  FOREIGN KEY(PNR_NUM) REFERENCES PASS_ACCOUNT(PNR),
  FOREIGN KEY(TRAIN_NUM) REFERENCES TRAIN(TRAIN_NO)
);|
```

**Results** Explain Describe Saved SQL History

Table created.

0.01 seconds

## 7. SEAT\_AVAILABILITY TABLE:

Home > SQL > SQL Commands

☒ Autocommit Display 10 ▼

```
CREATE TABLE SEAT_AVAILABILITY
(
  TR_NUM INT NOT NULL,
  DATE_RES DATE NOT NULL,
  CLASS VARCHAR(20) NOT NULL,
  QUOTA VARCHAR(20) NOT NULL,
  TOTAL_SEATS INT NOT NULL,
  SEATS_AVAIL INT NOT NULL,
  FOREIGN KEY(TR_NUM) REFERENCES TRAIN(TRAIN_NO)
)
```

**Results** Explain Describe Saved SQL History

Table created.

0.00 seconds

## ALTER TABLE COMMANDS:

Home > SQL > SQL Commands

☒ Autocommit Display 10 S

```
ALTER TABLE PASS_ACCOUNT  
ADD DATE_OF_BOOKING DATE;]
```

**Results** Explain Describe Saved SQL History

Table altered.

0.00 seconds

User: SYSTEM

Home > SQL > SQL Commands

☒ Autocommit Display 10

```
ALTER TABLE PASS_ACCOUNT  
DROP COLUMN DATE_OF_BOOKING]
```

**Results** Explain Describe Saved SQL History

Table altered.

0.00 seconds

## **DML Commands:**

DML stands for Data Manipulation Language. DML commands are used for insertion, deletion and modification of data in a database. The most commonly used DML commands are:

1. INSERT: This command is used to insert new records into a relation.

**Syntax:** INSERT INTO *table\_name*  
VALUES (*value1*, *value2*, *value3*  
, ....);

2. UPDATE: This command is used to update existing records in a relation.

**Syntax:** UPDATE *table\_name*  
SET *column1* = *value1*, *column2* = *value2*, ...  
WHERE *condition*;

3. DELETE: This command is used to delete desired records from a relation.

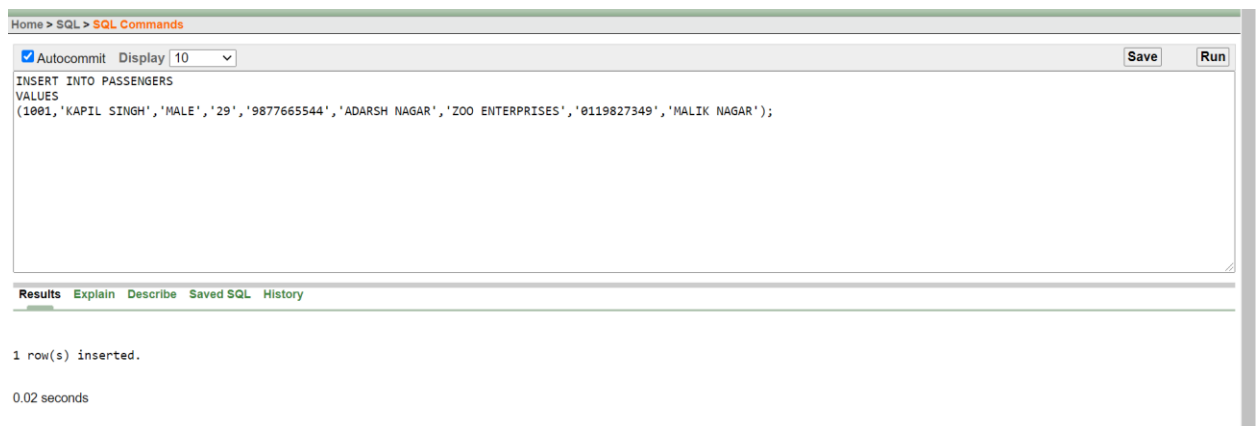
**Syntax:** DELETE FROM *table\_name* WHERE *condition*;

4. SELECT: This command is used to retrieve data from a database.

**Syntax:** SELECT *column1*, *column2*, ... FROM *table\_name*;

## Implementing DML Commands:

### 1. INSERT:



## 2. UPDATE:

Home > SQL > **SQL Commands**

☒ Autocommit   Display 10 ▾

UPDATE PASSENGERS SET AGE='23' WHERE PERSONAL\_ID = 1002;

Results   Explain   Describe   Saved SQL   History

1 row(s) updated.

0.00 seconds

## 3. DELETE:

Home > SQL > **SQL Commands**

☒ Autocommit   Display 10 ▾

DELETE FROM PASSENGERS WHERE PERSONAL\_ID = 1002;

Results   Explain   Describe   Saved SQL   History

1 row(s) deleted.

0.00 seconds

#### 4. SELECT:

Home > SQL > SQL Commands

☒ Autocommit Display 10

```
SELECT * FROM PASSENGERS
```

**Results** Explain Describe Saved SQL History

PERSONAL_ID	NAME	SEX	AGE	PHONE_NO	ADDRESS	OFFICE_NAME	OFFICE_PHONE	OFFICE_ADDRESS
1001	KAPIL SINGH	MALE	29	9877665544	ADARSH NAGAR	ZOO ENTERPRISES	0119827349	MALIK NAGAR
1002	NIHAL SINGH	MALE	39	9824455544	VIDESH NAGAR	HUSAIN ENTERPRISES	0123827349	SANT NAGAR
1003	NISHANT SINGH	MALE	19	9824455542	SWAROOP NAGAR	INAYAT ENTERPRISES	0113227349	NEHRU VIHAR
1004	ABHISHEK JHA	MALE	18	9977555420	ROOP NAGAR	SHOK ENTERPRISES	0123227349	HARSH VIHAR
1005	SAPNA DAESAI	MALE	18	9977555333	PATEL NAGAR	VENKATESH HOTEL	0123222349	CHOWDRI VIHAR

5 rows returned in 0.01 seconds [CSV Export](#)

#### CONCLUSION:

At the end of this experiment, various common DDL and DML commands were successfully implemented.



## EXPERIMENT 8

**AIM:** Performing Simple Queries.

### **THEORY:**

#### **SQL Queries:**

Queries are used in SQL to manipulate and retrieve data from a database as per requirements. This is achieved with the help of some constraints. In SQL, the SELECT command is used for efficient query processing. The data retrieved is presented in the form a result table and the results are called result-sets.

#### **Performing Simple Queries:**

- 1. Retrieve all records from train table where time duration is more than 1111 minutes.**

Home > SQL > **SQL Commands**

☒ Autocommit Display 10 ▼

SELECT \* FROM TRAIN WHERE TIME\_DURATION>1111

**Results** Explain Describe Saved SQL History

TRAIN_NO	TRAIN_NAME	TIME_DURATION	FIRST_STATION	LAST_STATION
102	SOUTH EXPRESS	1112	SHAH PURI	CHANDNI CHOWK
103	EAST EXPRESS	1113	KARAWAL NAGAR	NOIDA

2 rows returned in 0.00 seconds [CSV Export](#)

- 2. Retrieve all records from passengers table whose age is greater than 20**

☒ Autocommit Display 10 ▼

SELECT \* FROM PASSENGERS WHERE AGE>20

**Results** Explain Describe Saved SQL History

PERSONAL_ID	NAME	SEX	AGE	PHONE_NO	ADDRESS	OFFICE_NAME	OFFICE_PHONE	OFFICE_ADDRESS
1001	KAPIL SINGH	MALE	29	9877665544	ADARSH NAGAR	ZOO ENTERPRISES	0119827349	MALIK NAGAR
1002	NIHAL SINGH	MALE	23	9824455544	VIDESH NAGAR	HUSAIN ENTERPRISES	0123827349	SANT NAGAR

2 rows returned in 0.00 seconds [CSV Export](#)

### 3. Display the Passenger Id, Passenger Name and Number of Seats from Reservations Table where TrainNum=103

User: SYSTEM

Home > SQL > **SQL Commands**

☒ Autocommit Display 10 ▾

```
SELECT PASSENGER_ID, PASSENGER_NAME, NUMBER_OF_SEATS FROM RESERVATION WHERE TRAIN_NUM=103
```

---

**Results** Explain Describe Saved SQL History

PASSENGER_ID	PASSENGER_NAME	NUMBER_OF_SEATS
12234	KAMAL SINGH	13
12224	KAMA SINGH	17

2 rows returned in 0.00 seconds [CSV Export](#)

### 4. Display those records from Account Table where Amount is between 3200 and 5700

Home > SQL > **SQL Commands**

☒ Autocommit Display 10 ▾

```
SELECT * FROM PASS_ACCOUNT WHERE AMOUNT > 3200 AND AMOUNT < 5700
```

---

**Results** Explain Describe Saved SQL History

PNR	PASS_NAME	AMOUNT	DATE_OF_BOOKING
11112	ROHIT SINGH	3500	22-FEB-21
11113	HARDIP SINGH	4500	22-JAN-21
11114	PADIP SINGH	5500	12-JAN-21

3 rows returned in 0.02 seconds [CSV Export](#)

## 5. Find out the number of trains in operation on the basis of their class

Home > SQL > SQL Commands

☒ Autocommit Display 10 ▼

```
SELECT CLASS, COUNT(*) FROM SEAT_AVAILABILITY GROUP BY CLASS;
```

Results Explain Describe Saved SQL History

CLASS	COUNT(*)
1AC	1
3AC	1
2AC	1

3 rows returned in 0.01 seconds

[CSV Export](#)

## 6. Calculate the total number of seats in trains based on quota

Home > SQL > SQL Commands

☒ Autocommit Display 10 ▼

```
SELECT QUOTA, SUM(TOTAL_SEATS) FROM SEAT_AVAILABILITY GROUP BY QUOTA;
```

Results Explain Describe Saved SQL History

QUOTA	SUM(TOTAL_SEATS)
GENERAL	101
LADIES	120
TATKAL	100

3 rows returned in 0.00 seconds

[CSV Export](#)

7. Retrieve information of all bookings from Account table made between 2021-01-15 and 2021-04-15.

Home / SQL / SQL Commands

☒ Autocommit Display 10 ▼

```
SELECT * FROM PASS_ACCOUNT WHERE DATE_OF_BOOKING  
BETWEEN TO_DATE('2021-01-15','YYYY-MM-DD') AND  
TO_DATE('2021-04-15','YYYY-MM-DD');|
```

Results Explain Describe Saved SQL History

PNR	PASS_NAME	AMOUNT	DATE_OF_BOOKING
11111	VIRAT SINGH	2500	22-MAR-21
11112	ROHIT SINGH	3500	22-FEB-21
11113	HARDIP SINGH	4500	22-JAN-21

3 rows returned in 0.05 seconds

[CSV Export](#)

## **CONCLUSION:**

As observed in the experiment, we understood how to perform simple queries on a database. The significance of the SELECT command and the GROUP BY command was understood. Also, we understood how to use basic aggregate functions like COUNT, SUM AND AVERAGE in SQL.

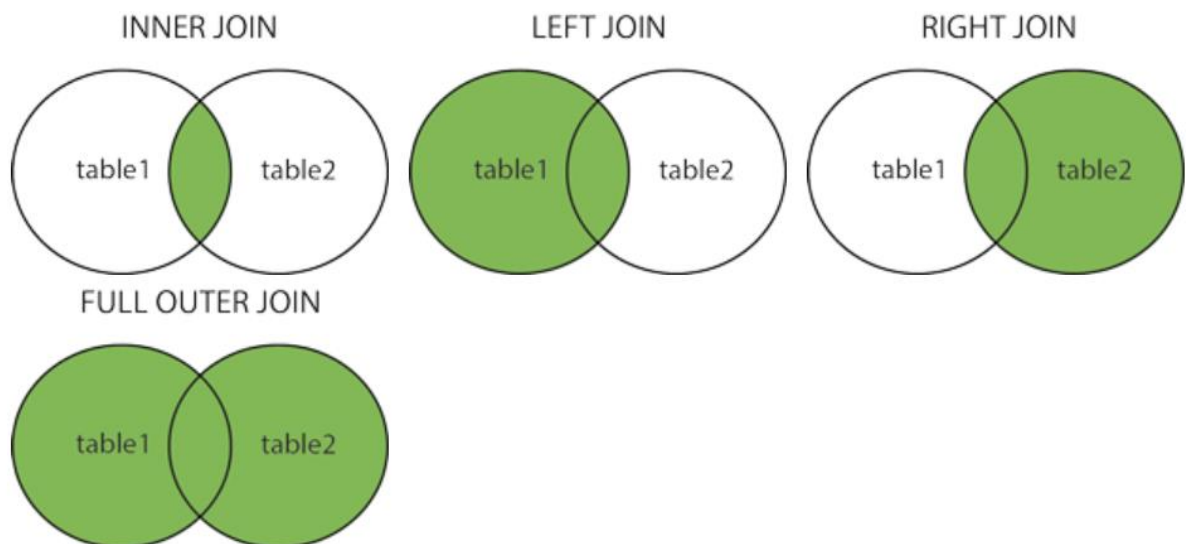
## **EXPERIMENT 9**

**AIM:** Implementation of Simple Joins

### **THEORY**

A Join is clause in SQL that is used to combine records from two or more tables based on a related attribute between them. There are five major joins in SQL. They are listed below.

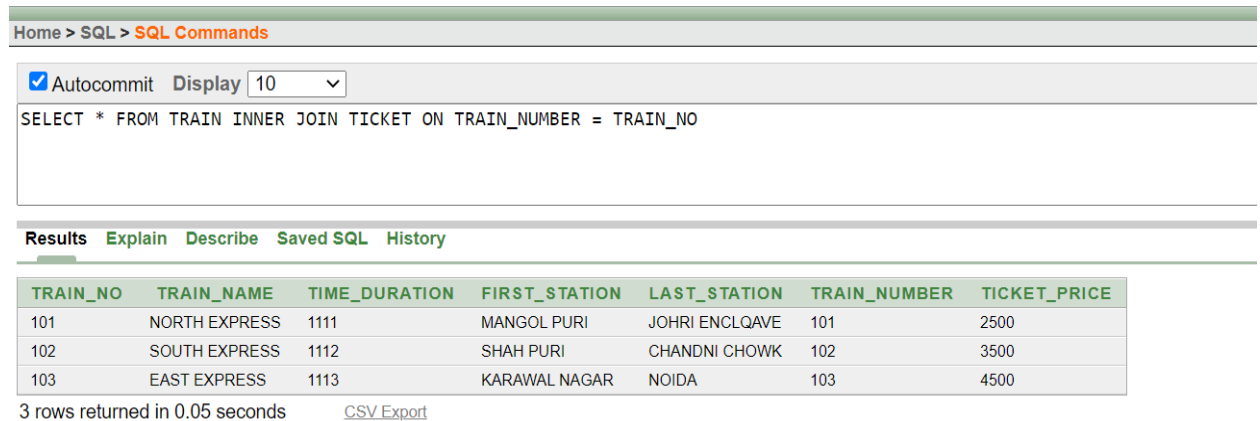
- **INNER JOIN:** This join is used to retrieve records that have matching values in both relations.
- **LEFT OUTER JOIN:** This join returns all records from the left relation and matching records from the right relation.
- **RIGHT OUTER JOIN:** This join returns all records from the right relation and matching records from the left relation.
- **FULL OUTER JOIN:** This join returns all the records along with their matches in left and right relation wherever they exist.
- **CROSS JOIN:** This join returns the Cartesian product of rows from the relations in the join.



## Implementation of joins:

- Inner Join

**Syntax:** SELECT columns FROM table1 INNER JOIN table2  
ON table1.column=table2.column WHERE Condition



The screenshot shows a web-based SQL interface. At the top, there's a breadcrumb 'Home > SQL > SQL Commands'. Below it, a toolbar includes a checked 'Autocommit' checkbox and a 'Display' dropdown set to '10'. The SQL query entered is 'SELECT \* FROM TRAIN INNER JOIN TICKET ON TRAIN\_NUMBER = TRAIN\_NO'. Below the query, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History', with 'Results' being the active tab. The results are displayed in a table with 7 columns: TRAIN\_NO, TRAIN\_NAME, TIME\_DURATION, FIRST\_STATION, LAST\_STATION, TRAIN\_NUMBER, and TICKET\_PRICE. Three rows of data are shown. Below the table, it states '3 rows returned in 0.05 seconds' and provides a 'CSV Export' link.

TRAIN_NO	TRAIN_NAME	TIME_DURATION	FIRST_STATION	LAST_STATION	TRAIN_NUMBER	TICKET_PRICE
101	NORTH EXPRESS	1111	MANGOL PURI	JOHRI ENCLQAVE	101	2500
102	SOUTH EXPRESS	1112	SHAH PURI	CHANDNI CHOWK	102	3500
103	EAST EXPRESS	1113	KARAWAL NAGAR	NOIDA	103	4500

3 rows returned in 0.05 seconds [CSV Export](#)

- Left Outer Join

**Syntax:** SELECT columns FROM table1 LEFT OUTER  
JOIN table2 ON table1.column=table2.column WHERE  
Condition

☒ Autocommit Display 10 ▼

```
SELECT PASS_ACCOUNT.PASS_NAME, CANCELLATION.NUMBER_OF_SEATS FROM PASS_ACCOUNT  
LEFT OUTER JOIN CANCELLATION ON PNR_NUM = PASS_ACCOUNT.PNR;|
```

**Results** Explain Describe Saved SQL History

PASS_NAME	NUMBER_OF_SEATS
VIRAT SINGH	12
ROHIT SINGH	11
HARDIP SINGH	16
PADIP SINGH	-

4 rows returned in 0.02 seconds

[CSV Export](#)

- **Right Outer Join**

**Syntax:** SELECT columns FROM table1 RIGHT OUTER  
JOIN table2 ON table1.column=table2.column WHERE  
Condition

☒ Autocommit Display 10 ▼

```
SELECT TRAIN.TRAIN_NAME, SEAT_AVAILABILITY.QUOTA, SEAT_AVAILABILITY.SEATS_AVAIL  
FROM TRAIN RIGHT OUTER JOIN SEAT_AVAILABILITY  
ON TR_NUM=TRAIN.TRAIN_NO WHERE SEAT_AVAILABILITY.SEATS_AVAIL > 20
```

**Results** Explain Describe Saved SQL History

TRAIN_NAME	QUOTA	SEATS_AVAIL
SOUTH EXPRESS	GENERAL	40
EAST EXPRESS	LADIES	30

2 rows returned in 0.02 seconds

[CSV Export](#)

- **Full Outer Join**

**Syntax:** SELECT columns FROM table1 FULL OUTER  
JOIN table2 ON table1.column=table2.column

UDM\_210116

Home > SQL > SQL Commands

☒ Autocommit Display 10 Save Run

```
SELECT *FROM PASSENGERS FULL OUTER JOIN RESERVATION ON PASSENGER_ID = PASSENGERS.PERSONAL_ID;
```

**Results** Explain Describe Saved SQL History

PERSONAL_ID	NAME	SEX	AGE	PHONE_NO	ADDRESS	OFFICE_NAME	OFFICE_PHONE	OFFICE_ADDRESS	RESERVATION_NO	PASSENGER_ID	PASSENGER_NAME	TRAIN_NUM	NUMBER_OF_SEATS	RESERVATION_DATE
1005	SAPNA DAESAI	MALE	18	9977555333	PATEL NAGAR	VENKATESH HOTEL	0123222349	CHOWDRI VIHAR	1122	1005	SAPNA DAESAI	101	12	21-MAR-21
1001	KAPIL SINGH	MALE	29	9977665544	ADARSH NAGAR	ZOO ENTERPRISES	0119927349	MALIK NAGAR	1123	1001	KAPIL SINGH	102	13	22-MAR-21
1004	ABHISHEK JHA	MALE	18	9977555420	ROOP NAGAR	SHOK ENTERPRISES	0123227349	HARSH VIHAR	1124	1004	ABHISHEK JHA	103	13	23-MAR-21
1003	NISHANT SINGH	MALE	19	9924455542	SIHARPOO NAGAR	INAYAT ENTERPRISES	0113227349	NEHRU VIHAR	1127	1003	NISHANT SINGH	103	17	24-MAR-21

4 rows returned in 0.01 seconds [CSV Export](#)

**Conclusion:** We understood how to perform various types of joins in a database and how they work.