DELHI TECHNOLOGICAL UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CO 202: DATABASE MANAGEMENT SYSTEMS

PRACTICAL FILE

Submitted To:	**Submitted By:**
Ms. Indu Singh	**Neeraj Sharma**
Assistant Professor	**2K19/CO/244**
Department of CSE, DTU	**A4 Batch Group-1**

TABLE OF CONTENTS

S.NO	Title	Date	Page No.
1.	Introduction to SQL, Database, and Database Management System.	January 18, 2021	4-5
2.	Introduction to Various software of DBMS and Database Languages.	January 25, 2021	6-10
3.	Introduction to ER Diagram and Symbol table.	February 8, 2021	11-14
4.	Introduction to Different types of constraints in SQL	February 22, 2021	15-17
5	Case Study 1: Railway Management System	March 12, 2021	18-19
6	Creating Entity-Relationship Diagram, Entity Tables and Relationship Tables	Match 12, 2021	20-22
7.	Implementing DDL and DML commands.	March 22, 2021	23-32
8.	Performing Simple Queries.	March 29, 2021	33-37

9.	Implementation of Simple Joins	March 31, 2021	38-41
10.	Implementing Aggregate functions, Date functions, Time functions.	April 27, 2021	42-48
11.	Implement Grant and Revoke statements, Views and the Triggers	April 29, 2021	49-55

EXPEREMENT NO – 1

AIM: Introduction to SQL, DATABASE and DATABASE MANAGEMENT SYSTEM.

THEORY:

1. **SQL:**

   - It is a database computer language designed for the retrieval and management of data in a relational database. **SQL** stands for **Structured Query Language.**

   - It is the standard language for Relational Database System. All the Relational Database Management Systems (RDMS) like MySQL, MS Access, Oracle, Sybase, Informix, Postgres and SQL Server use SQL as their standard database language.

   - It allows users to query the database in number of ways, using English-like statements.

Some of **SQL** commands are:

- **SELECT:** It extracts data from database.
- **INSERT INTO**: It inserts new data into a database.
- **UPDATE**: It updates data in a database.
- **DELETE**: It deletes data from database.
- **CREATE DATABASE**: It creates a new database.
- **ALTER DATABASE**: It modifies a database.
- **CREATE TABLE**: It create a new table.
- **ALTER TABLE**: It modifies the table.
- **DROP TABLE**: It deletes a Table.
- **CREATE INDEX**: It creates an index (search key).
- **DROP INDEX**: It deletes an index.

2. **DATABASE:**

   - A database is a data structure that stores organized information. Most databases contain multiple tables, which may each include several different fields. For example, a company database may include tables for products, employees, and financial records. Each of these tables would have different fields that are relevant to the information stored in the table.

- Nearly all e-commerce sites use the databases to store product inventory and customer information. These sites use a database management system (or DBMS), such as Microsoft Access, FileMaker Pro, or MySQL as the "back end" to the website. By storing website data in a database, the data can be easily searched, sorted, and updated. This flexibility is important for e-commerce sites and other types of dynamic websites.

3. **DATABASE MANAGEMENT SYSTEM:**
   - A Database Management System is a software that is designed to help create and maintain databases. It allows for the efficient storage of data. Popular DBMS software's are My SQL, Oracle, FoxPro etc. Some of its common functions are: o Data Dictionary Management: DBMS stores description about the data in the data dictionary and uses it to look up the required structures and relationships.

   - Data Storage Management: It efficiently stores all data in the database and manages it by itself. The user doesn't need to know how the data is being stored.

     - Efficient Query Processing: DBMS allows efficient query processing as queries can be executed in simple English-like syntaxes.
     - Restrict Unauthorized Access: It uses an Access Control Matrix to manage to maintain data integrity and prevent any intrusions.

Learning: In this particular experiment I learned SQL, a lot of SQL commands and about DBMS.

CONCLUSION:

This experiment introduces us to the concepts of database, DBMS and SQL. It also gives us a small outlook on SQL commands.

EXPEREMENT NO – 2

AIM: Introduction to various software of Database Management System and Database Language.

THEORY:

1. **DBMS** SOFTWARE:
    - MySQL
    - Microsoft Access
    - Oracle
    - PostgreSQL
    - dbase
    - FoxPro
    - SQLite
    - IBM DB2
    - LibreOffice Base
    - MariaDB
    - Microsoft SQL Server
    - MongoDB

- **MYSQL:**

  MySQL is the most popular Open Source Relational SQL Database Management System. MySQL is one of the best RDBMS being used for developing various web-based software applications. MySQL is developed, marketed and supported by MySQL AB, which is a Swedish company.

- **SQLite:**

  SQLite is an in-process library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. It is a database, which is zero-configured, which means like other databases you do not need to configure it in your system.

  SQLite engine is not a standalone process like other databases, you can link it statically or dynamically as per your requirement with your application. SQLite accesses its storage files directly.

- **Oracle:**

  It is the most widely used DBMS software in the world. It supports multiple Linux, Unix and Windows versions. The latest version, Oracle 12c supports cloud services. It is a relational database management software. It is secured, occupies less space, supports large databases, and reduces CPU time to process data

- **MongoDB:**
  MongoDB is an open-source document database and leading NoSQL database. MongoDB is written in C++. This tutorial will give you great understanding on MongoDB concepts needed to create and deploy a highly scalable and performance-oriented database.

- **Microsoft Access:**

  It is a DBMS software that uses Graphical User Interface (GUI) and software development tools. It is considered ideal for beginners to learn basics of database due to its use of GUI. It is used by most e-commerce websites. It orks only on Microsoft Windows.

- **Microsoft SQL Server:**

  Developed by Microsoft in 1989, it works on both Linux and Windows Operating Systems. The language used is Assembly C, Linux, C++ for writing it. It is compatible with Oracle, provides efficient management of workload and allows multiple users to use the same database.

2. **Database Languages:**

   Database Languages are used to perform various types of operations on a database. It is used to define and manipulate a database. It is of five major types. They are:

   - **Data Manipulation Language:**
     A data manipulation language (DML) is a family of computer languages including commands permitting users to manipulate data in a database. This manipulation involves inserting data into database tables, retrieving existing data, deleting data from existing tables and modifying existing data. DML is mostly incorporated in SQL databases.

     DML resembles simple English language and enhances efficient user interaction with the system. The functional capability of DML is organized in manipulation commands like SELECT, UPDATE, INSERT INTO and DELETE FROM, as described below:

     - SELECT: This command is used to retrieve rows from a table. The syntax is SELECT [column name(s)] from [table name] where [conditions]. SELECT is the most widely used DML command in SQL.
     - UPDATE: This command modifies data of one or more records. An update command syntax is UPDATE [table name] SET [column name = value] where [condition].

- INSERT: This command adds one or more records to a database table. The insert command syntax is INSERT INTO [table name] [column(s)] VALUES [value(s)].
- DELETE: This command removes one or more records from a table according to specified conditions. Delete command syntax is DELETE FROM [table name] where [condition].

❖ Data Definition Language:

- **DDL** stands for **D**ata **D**efinition **L**anguage. It is used to define database structure or pattern.
- It is used to create schema, tables, indexes, constraints, etc. in the database.
- Using the DDL statements, you can create the skeleton of the database.
- Data definition language is used to store the information of metadata like the number of tables and schemas, their names, indexes, columns in each table, constraints, etc.

Here are some tasks that come under DDL:

- **Create:** It is used to create objects in the database.
- **Alter:** It is used to alter the structure of the database.
- **Drop:** It is used to delete objects from the database.
- **Truncate:** It is used to remove all records from a table.
- **Rename:** It is used to rename an object.
- **Comment:** It is used to comment on the data dictionary.

These commands are used to update the database schema that's why they come under Data definition language.

❖ Data Control Language:

- **DCL** stands for **D**ata **C**ontrol **L**anguage. It is used to retrieve the stored or saved data.
- The DCL execution is transactional. It also has rollback parameters.

(But in Oracle database, the execution of data control language does not have the feature of rolling back.)

Here are some tasks that come under DCL:

- **Grant:** It is used to give user access privileges to a database.
- **Revoke:** It is used to take back permissions from the user.

There are the following operations which have the authorization of Revoke:

CONNECT, INSERT, USAGE, EXECUTE, DELETE, UPDATE and SELECT.

❖ Transaction Control Language:

TCL is used to run the changes made by the DML statement. TCL can be grouped into a logical transaction.

Here are some tasks that come under TCL:

- ❖ **Commit:** It is used to save the transaction on the database.
- ❖ **Rollback:** It is used to restore the database to original since the last Commit.

❖ Session Control Language:

- Session Control Language: It is referred to as SCL. It is used to dynamically manage the properties of a session. The SCL commands are as follows. o Alter Session: It is used to modify conditions or parameters affecting database connections.
- o Set Role: It is used to enable or disable the roles that are currently enabled for the session.

Learning: In this particular experiment we learned different types of softwares for DBMS And Database Language.

CONCLUSION:

This experiment introduced us to various DBMS software, about various Data Languages. Various SQL commands were briefly discussed.

EXPEREMENT NO – 3

AIM: Introduction to Entity Relationship (ER) Diagram, Symbol Table.

THEORY:

ER DIAGRAM:

ER Model stands for Entity Relationship Model is a high-level conceptual data model diagram. ER model helps to systematically analyze data requirements to produce a well-designed database. The ER Model represents real-world entities and the relationships between them. Creating an ER Model in DBMS is considered as a best practice before implementing your database.

ER Modeling helps you to analyze data requirements systematically to produce a well-designed database. So, it is considered a best practice to complete ER modeling before implementing your database.

ER diagrams are a visual tool which is helpful to represent the ER model. It was proposed by Peter Chen in 1971 to create a uniform convention which can be used for relational database and network. He aimed to use an ER model as a conceptual modeling approach.

Why use ER Diagrams?

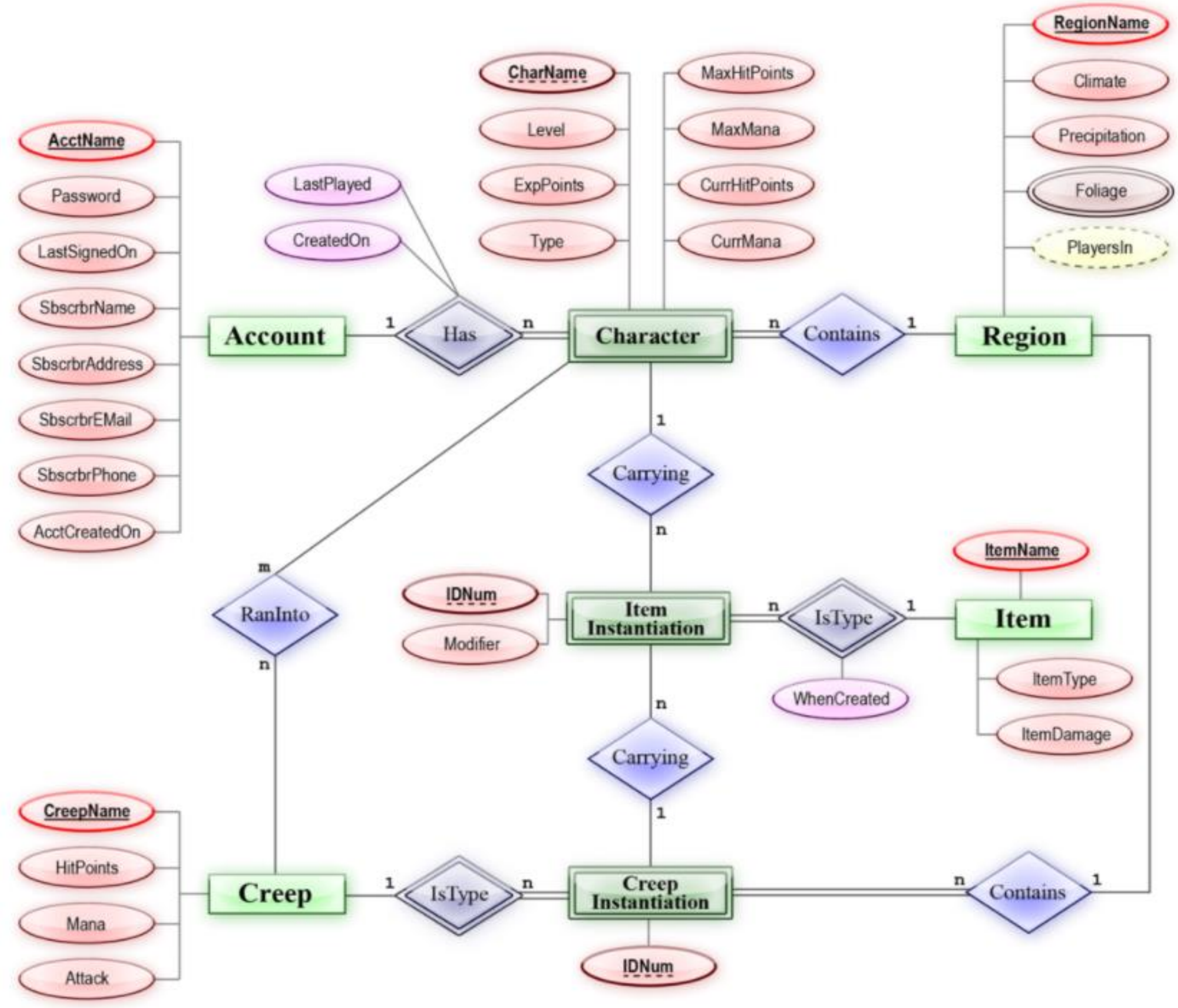
Here, are prime reasons for using the ER Diagram

- Helps you to define terms related to entity relationship modeling
- Provide a preview of how all your tables should connect, what fields are going to be on each table
- Helps to describe entities, attributes, relationships
- ER diagrams are translatable into relational tables which allows you to build databases quickly
- ER diagrams can be used by database designers as a blueprint for implementing data in specific software applications
- The database designer gains a better understanding of the information to be contained in the database with the help of ERP diagram
- ERD Diagram allows you to communicate with the logical structure of the database to users

Disadvantages of using an ER diagram:

□ Loss of information content: Some information be lost or hidden in ER model □ Limited relationship representation: ER model represents limited relationship as compared to another data models like relational model etc.

□ No representation of data manipulation: It is difficult to show data manipulation in ER model.



Symbol Tables: Symbol table is an important data structure created and maintained by compilers in order to store information about the occurrence of various entities such as variable names, function names, objects, classes, interfaces, etc.

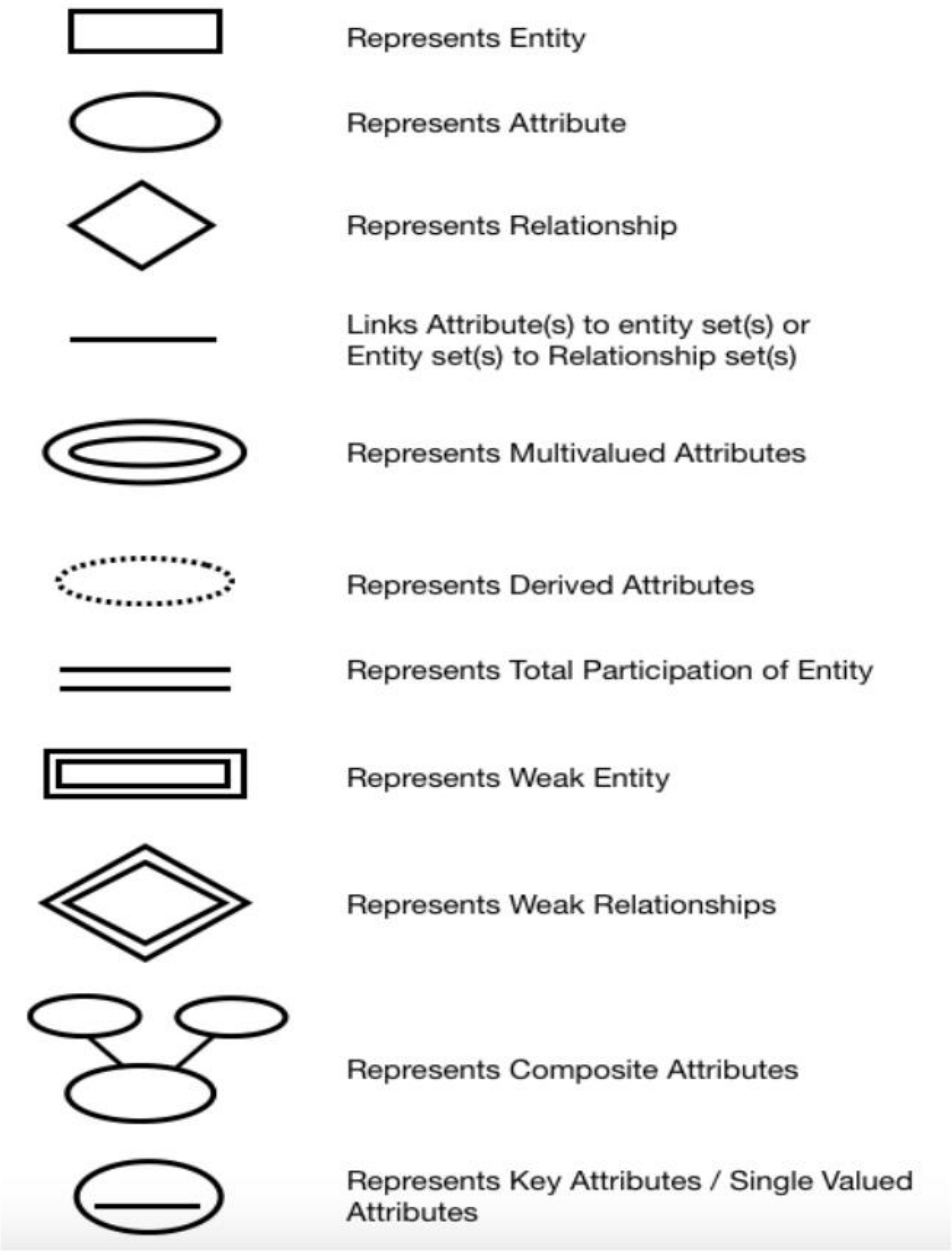

Represents Entity
Represents Attribute
Represents Relationship
Links Attribute(s) to entity set(s) or Entity set(s) to Relationship set(s)
Represents Multivalued Attributes
Represents Derived Attributes
Represents Total Participation of Entity
Represents Weak Entity
Represents Weak Relationships
Represents Composite Attributes
Represents Key Attributes / Single Valued Attributes

Figure: Symbols used in ER diagram

Following are the main components and its symbols in ER Diagrams:

- **Rectangles:** This Entity Relationship Diagram symbol represents entity types
- **Ellipses:** Symbol represent attributes
- **Diamonds:** This symbol represents relationship types
- **Lines:** It links attributes to entity types and entity types with other relationship types
- **Primary key:** attributes are underlined
- **Double Ellipses:** Represent multi-valued attributes

Learning: In this particular experiment we learned about the ER diagram and symbol table.

CONCLUSION:

In this experiment, the concept of ER Diagrams and Symbol Table was tried to explained. The various symbols used to represent various components of an ER diagram.

EXPEREMENT NO – 4

AIM: Introduction to different types of Constraints in SQL.

THEORY:

Constraints are the rules that we can apply on the type or data in a table. That is, we can specify the limit on type of data can be stored in a particular column in a table using constraints.

The available constraints in SQL are:

- **NOT NULL:** This constraint tells that we cannot store a NULL value in a
- column. That is, if a column is specified as NOT NULL then we will not be able
- to store null in this particular column anymore.
- **UNIQUE:** This constraint when specified with a column, tells that all the values
- in the column must be unique. That is, the value in any row of a column must not
- be repeated.
- **PRIMARY KEY:** A primary key is a field which can uniquely identify each row
- in a table. And this constraint is used to specify a field in a table as primary key.
- **FOREIGN KEY:** A Foreign key is a field which can uniquely identify each row
- in another table. And this constraint is used to specify a field as Foreign key.
- **CHECK:** This constraint helps to validate the values of a column to meet a
- particular condition. That is, it helps to ensure that he value stored in a column
- meets a specific condition.
- **DEFAULT:** This constraint specifies a default value for the column when no
- value in specified by user.

NOT NULL CONSTRAINT:

By default, a column can hold NULL values. If you do not want a column to have a NULL value, then you need to define such a constraint on this column specifying that NULL is now not allowed for that column.

Example

For example, the following SQL query creates a new table called CUSTOMERS and adds five columns, three of which, are ID NAME and AGE, In this we specify not to accept NULLs −

```
CREATE TABLE CUSTOMERS (
  ID   INT            NOT NULL,
  NAME VARCHAR (20)     NOT NULL,
  AGE INT             NOT NULL,
  ADDRESS CHAR (25),
  SALARY   DECIMAL (18, 2),
  PRIMARY KEY (ID)
);
```

UNIQUE CONSTRAINT:

he UNIQUE constraint uniquely identifies each record in a database table.

The UNIQUE and PRIMARY KEY constraints both provide a guarantee for uniqueness for a column or set of columns.

For eg.

```
CREATE TABLE Persons
(
P_Id int NOT NULL UNIQUE,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255)
)
```

DEFAULT **CONSTRAINT**: It is used to provide a default value to a column when none is specified. For example, country of students could be given a default value. CREATE TABLE Students (Roll_No. int NOT NULL UNIQUE, Name varchar(255) NOT NULL, Address varchar(255), Age int, Country varchar(255) DEFAULT 'India');

PRIMARY KEY CONSTRAINT:

The PRIMARY KEY constraint uniquely identifies each record in a table.

Primary keys must contain UNIQUE values, and cannot contain NULL values.

A table can have only ONE primary key; and in the table, this primary key can consist of single or multiple columns (fields).

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    PRIMARY KEY (ID)
);
```

CHECK CONSTRAINT:

It is used to ensure that a certain condition is met when a record is entered into a relation. For example: students studying in a school should not be more than 25 years old.

```
CREATE TABLE Students ( Roll_No. int NOT NULL UNIQUE PRIMARY KEY,
Name varchar(255) NOT NULL, Address varchar(255), Age int NOT NULL CHECK
(age<=25), Country varchar(255) DEFAULT 'India'
);
```

FOREIGN KEY CONSTRAINT: A It is used to indicate a row that uniquely identifies a record in another table. For
example: department id acts as foreign key in employees table which acts as primary key
in departments table.

```
CREATE TABLE Employees (
emp_id int NOT NULL UNIQUE PRIMARY KEY,
Name varchar(255) NOT NULL,
salary int,
dept_id int,
FOREIGN KEY(dept_id) REFERENCES departments(dept_id)
);
```

CONCLUSION:

We learned SQL contains multiple constraints and we also learned how to use them.

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

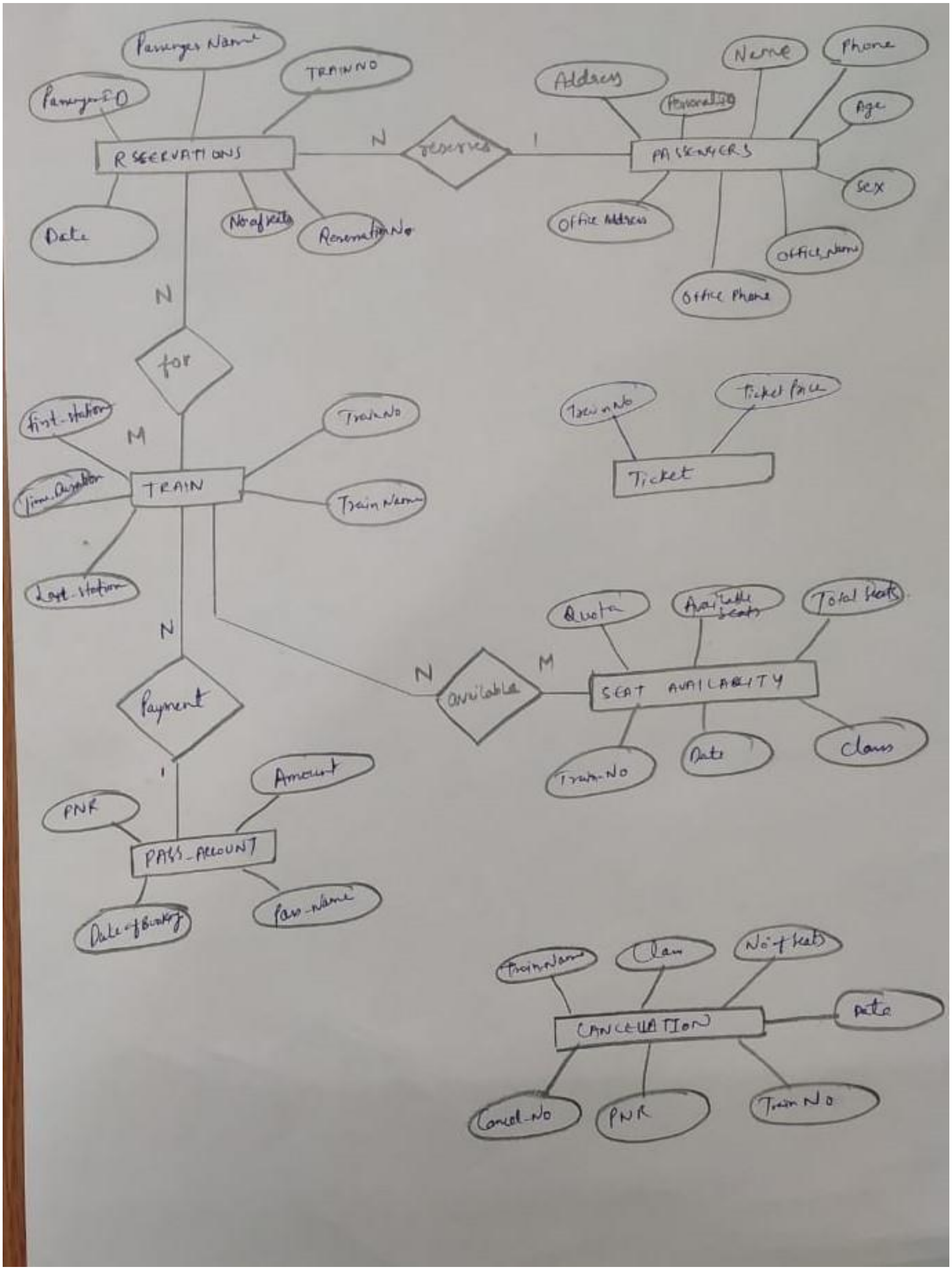
THEORY

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.


Passenger Name
TRAIN NO
Passenger ID
RESERVATIONS
Date
No of seats
Reservation No
N
reserves
1
Address
Personal ID
Name
Phone
Age
PASSENGERS
Sex
Office Address
Office_Name
Office Phone
N
for
M
first_station
Train No
Time_Duration
TRAIN
Train Name
Last_station
Train No
Ticket Price
Ticket
Quota
Available Seats
Total Seats
N
N
available
M
SEAT AVAILABILITY
Payment
Train_No
Date
Class
1
Amount
PNR
PASS_ACCOUNT
Pass_name
Date of Booking
Train Name
Class
No of Seats
Date
CANCELLATION
Cancel_No
PNR
Train No

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 Attibute_Name datatype;

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

   Syntax: DROP TABLE table_name;

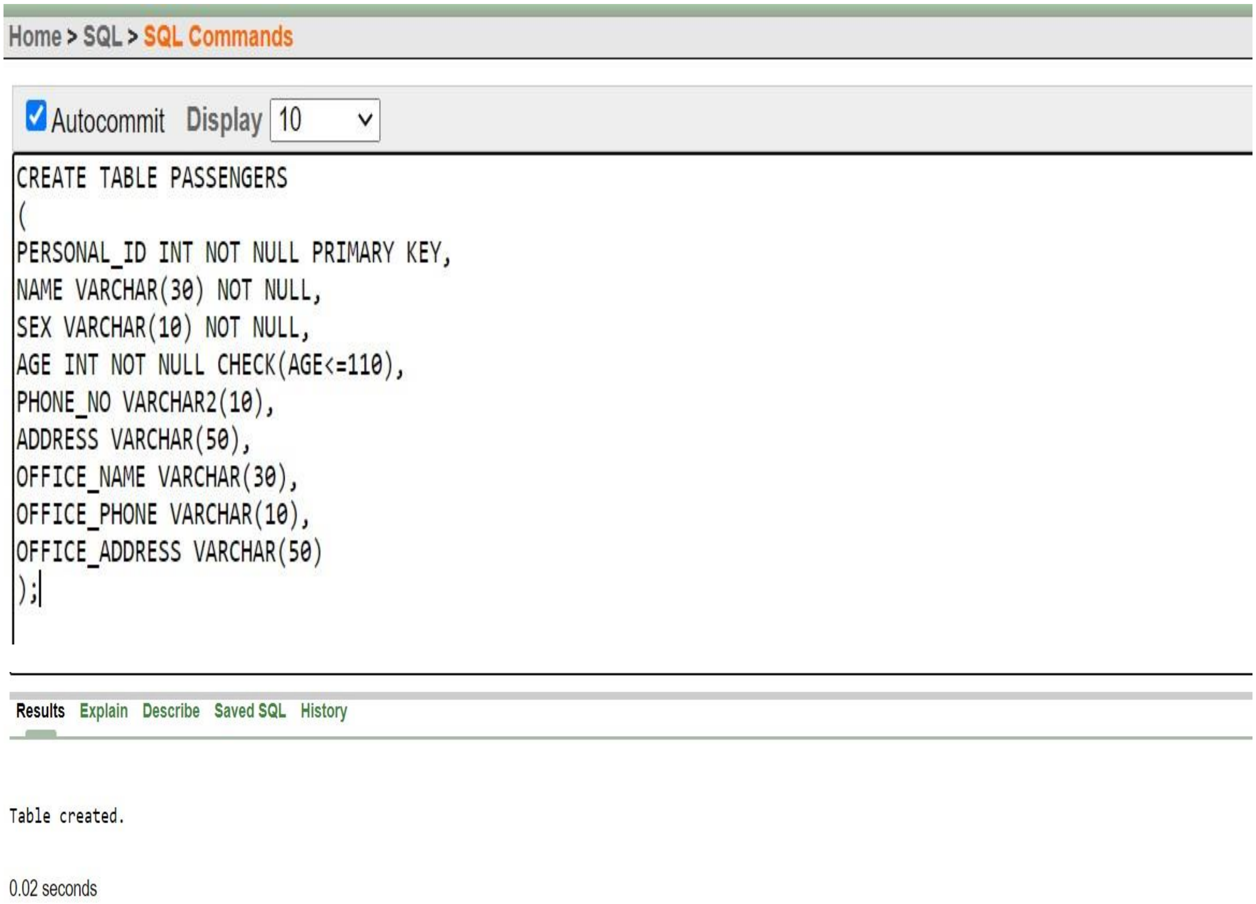
4. **TRUNCATE:** This command is used to erase complete data from an existing relation **Syntax:** TRUNCATE TABLE table_name;

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:

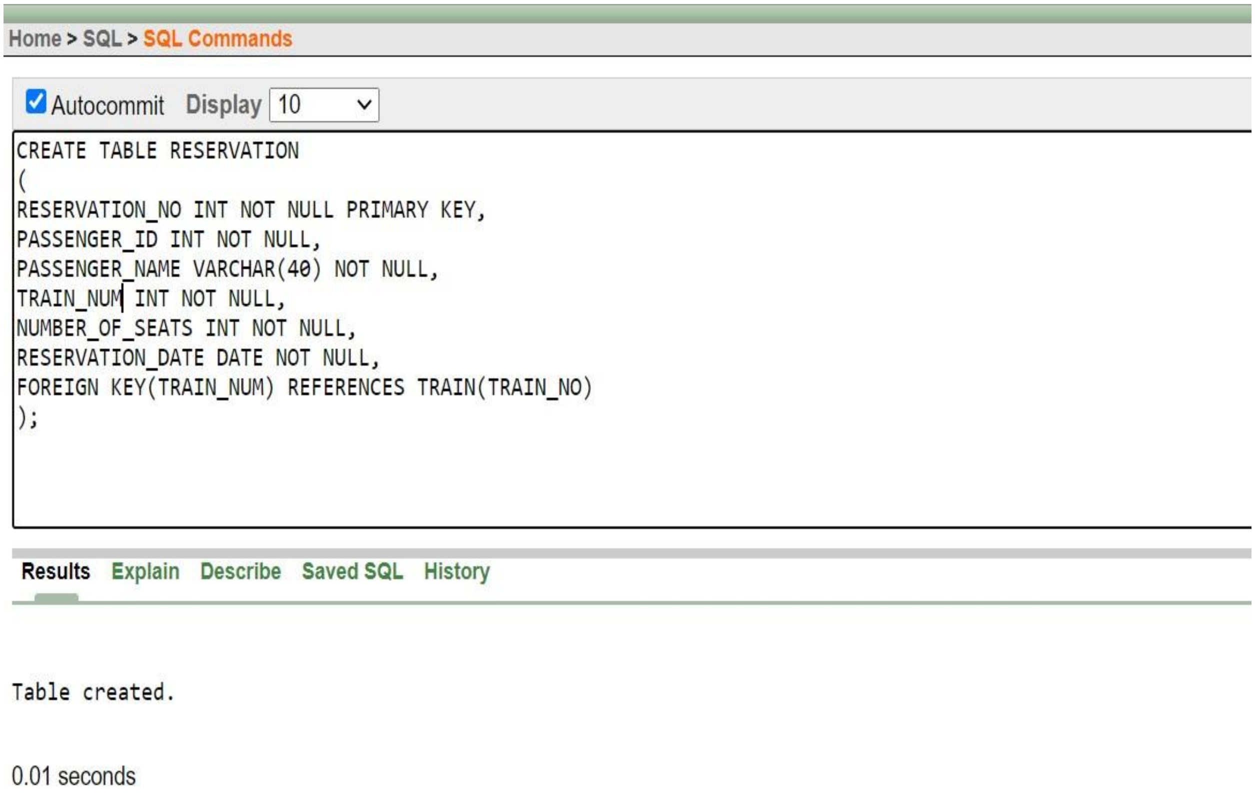


```
CREATE TABLE PASSENGERS
(
PERSONAL_ID INT NOT NULL PRIMARY KEY,
NAME VARCHAR(30) NOT NULL,
SEX VARCHAR(10) NOT NULL,
AGE INT NOT NULL CHECK(AGE<=110),
PHONE_NO VARCHAR2(10),
ADDRESS VARCHAR(50),
OFFICE_NAME VARCHAR(30),
OFFICE_PHONE VARCHAR(10),
OFFICE_ADDRESS VARCHAR(50)
);
```

```
Table created.

0.02 seconds
```

2. RESERVATION TABLE:

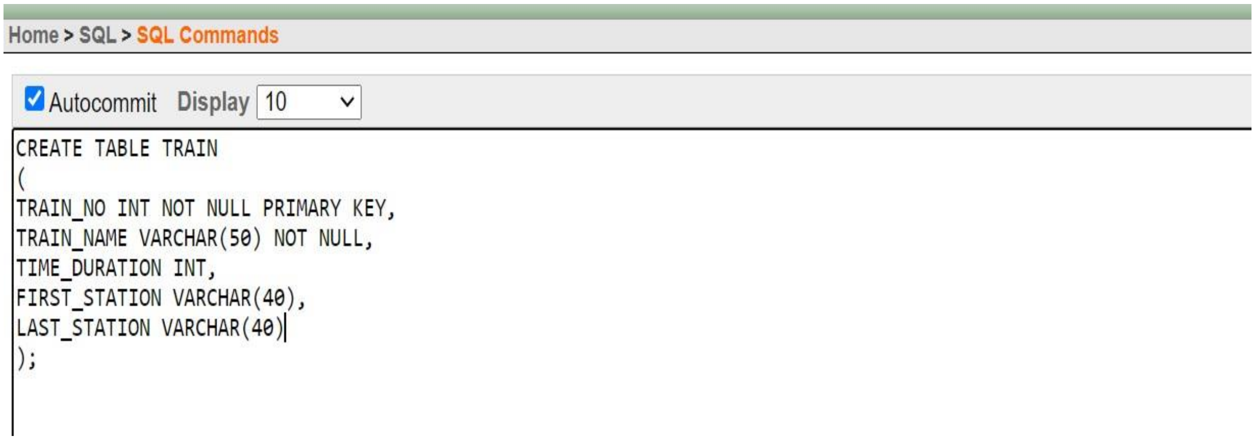


```
CREATE TABLE RESERVATION
(
RESERVATION_NO INT NOT NULL PRIMARY KEY,
PASSENGER_ID INT NOT NULL,
PASSENGER_NAME VARCHAR(40) NOT NULL,
TRAIN_NUM INT NOT NULL,
NUMBER_OF_SEATS INT NOT NULL,
RESERVATION_DATE DATE NOT NULL,
FOREIGN KEY(TRAIN_NUM) REFERENCES TRAIN(TRAIN_NO)
);
```

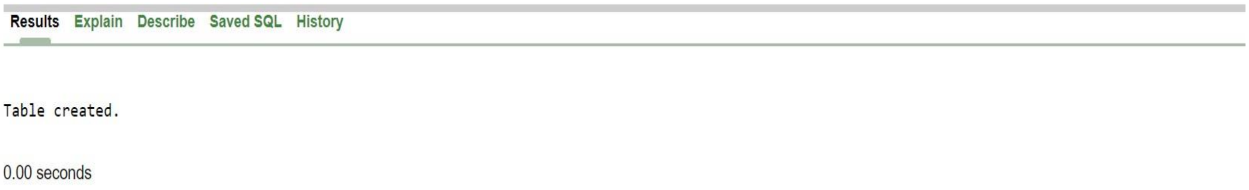
```
Table created.

0.01 seconds
```

3. TRAIN TABLE:



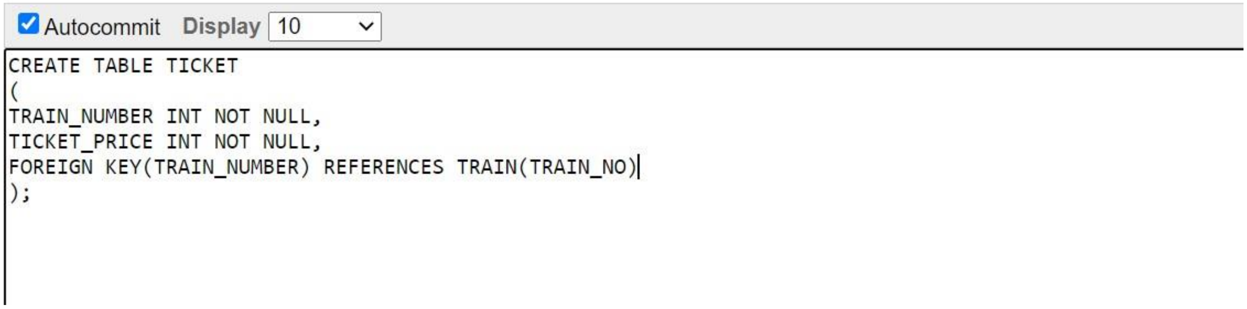
```
CREATE TABLE TRAIN
(
TRAIN_NO INT NOT NULL PRIMARY KEY,
TRAIN_NAME VARCHAR(50) NOT NULL,
TIME_DURATION INT,
FIRST_STATION VARCHAR(40),
LAST_STATION VARCHAR(40)
);
```



```
Table created.
```

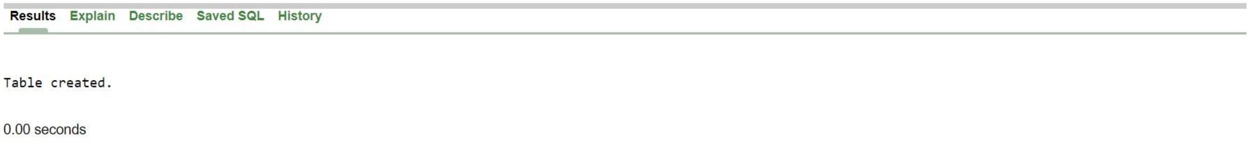
0.00 seconds

4. TICKET TABLE:



☑ Autocommit Display 10

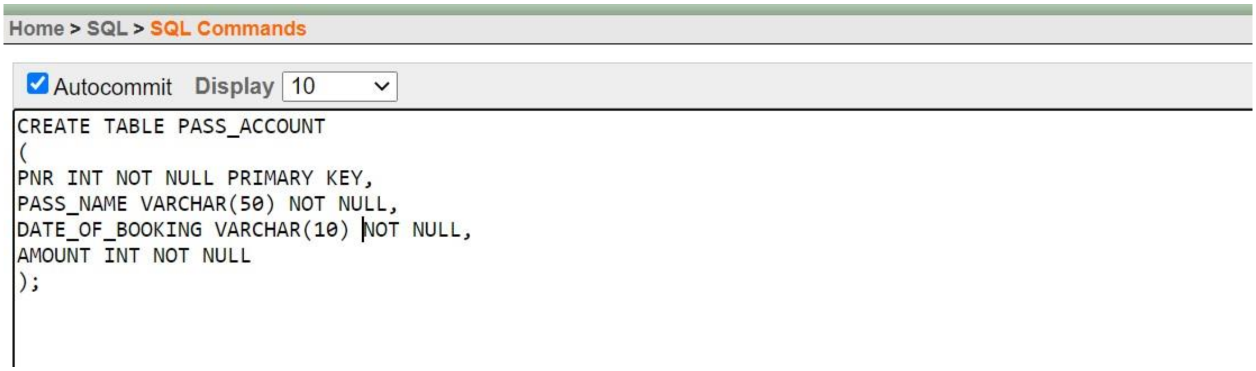
```
CREATE TABLE TICKET
(
TRAIN_NUMBER INT NOT NULL,
TICKET_PRICE INT NOT NULL,
FOREIGN KEY(TRAIN_NUMBER) REFERENCES TRAIN(TRAIN_NO)
);
```



```
Table created.
```

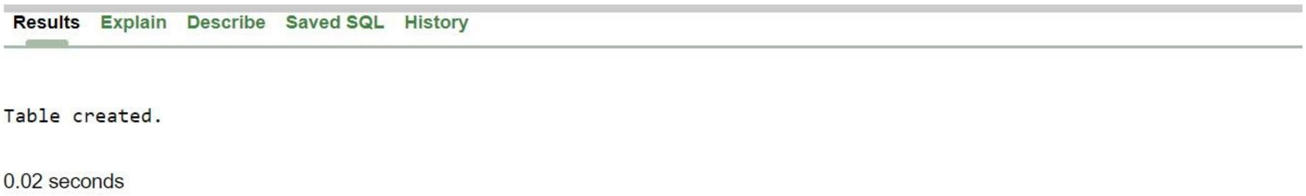
0.00 seconds

5. PASS_ACCOUNT TABLE:



☑ Autocommit Display 10

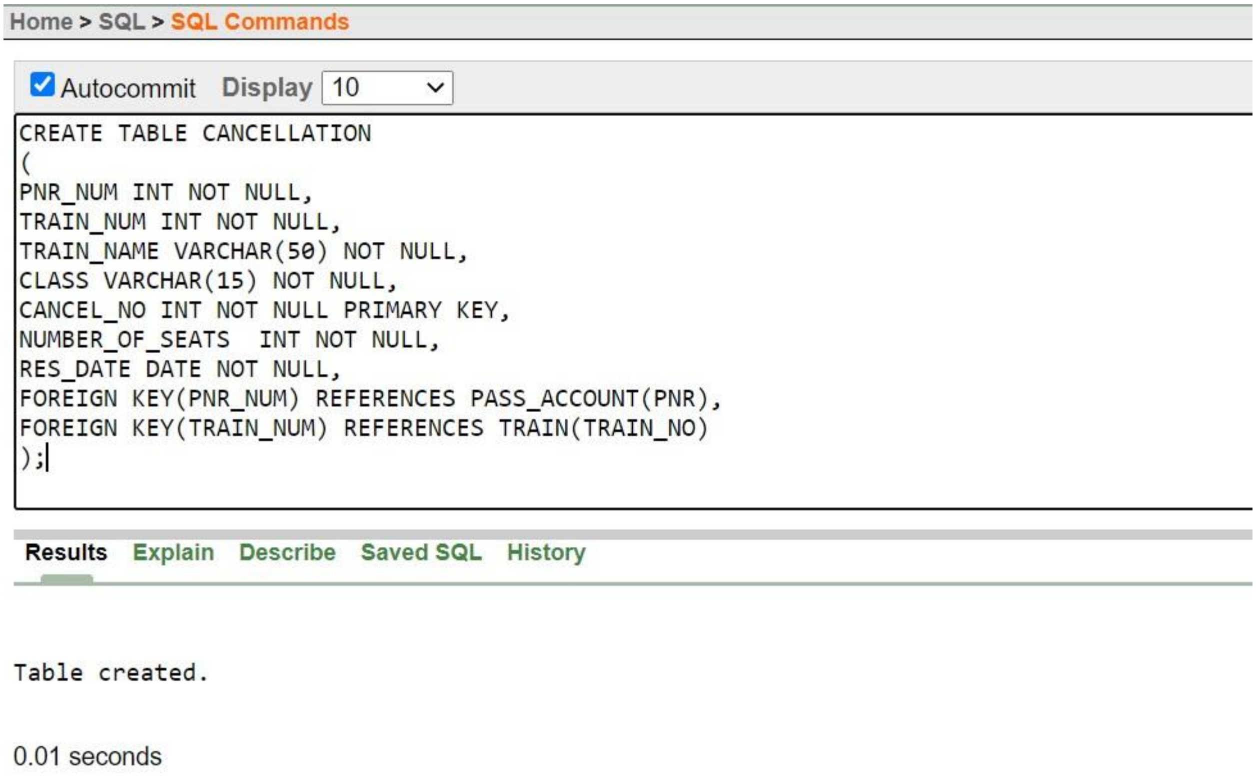
```
CREATE TABLE PASS_ACCOUNT
(
PNR INT NOT NULL PRIMARY KEY,
PASS_NAME VARCHAR(50) NOT NULL,
DATE_OF_BOOKING VARCHAR(10) NOT NULL,
AMOUNT INT NOT NULL
);
```



```
Table created.
```

0.02 seconds

6. CANCELLATION TABLE:

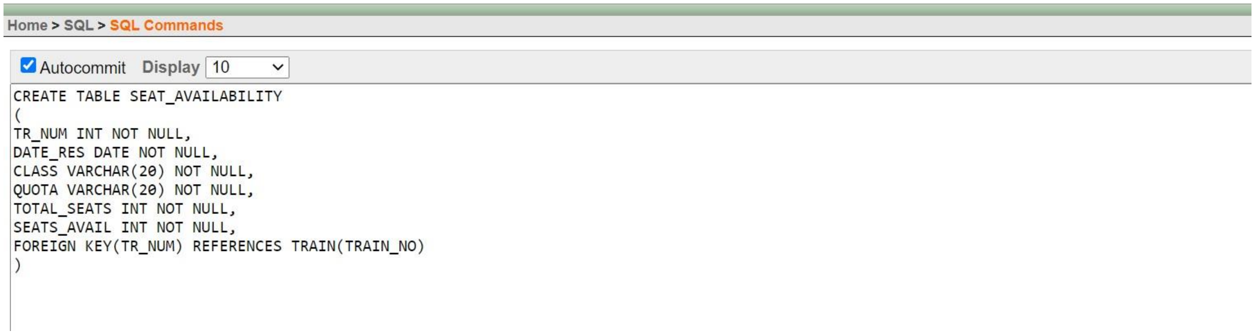


```
CREATE TABLE CANCELLATION
(
PNR_NUM INT NOT NULL,
TRAIN_NUM INT NOT NULL,
TRAIN_NAME VARCHAR(50) NOT NULL,
CLASS VARCHAR(15) NOT NULL,
CANCEL_NO INT NOT NULL PRIMARY KEY,
NUMBER_OF_SEATS  INT NOT NULL,
RES_DATE DATE NOT NULL,
FOREIGN KEY(PNR_NUM) REFERENCES PASS_ACCOUNT(PNR),
FOREIGN KEY(TRAIN_NUM)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

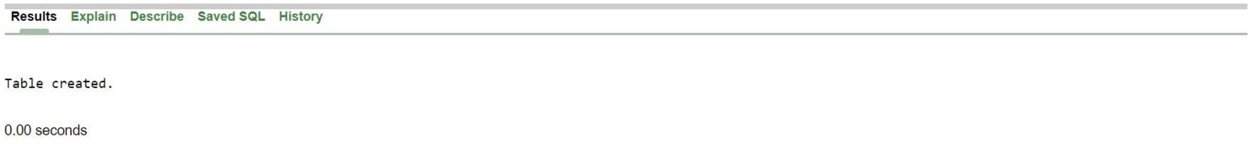
```
Table created.

0.01 seconds
```

7. SEAT_AVAILABILITY TABLE:



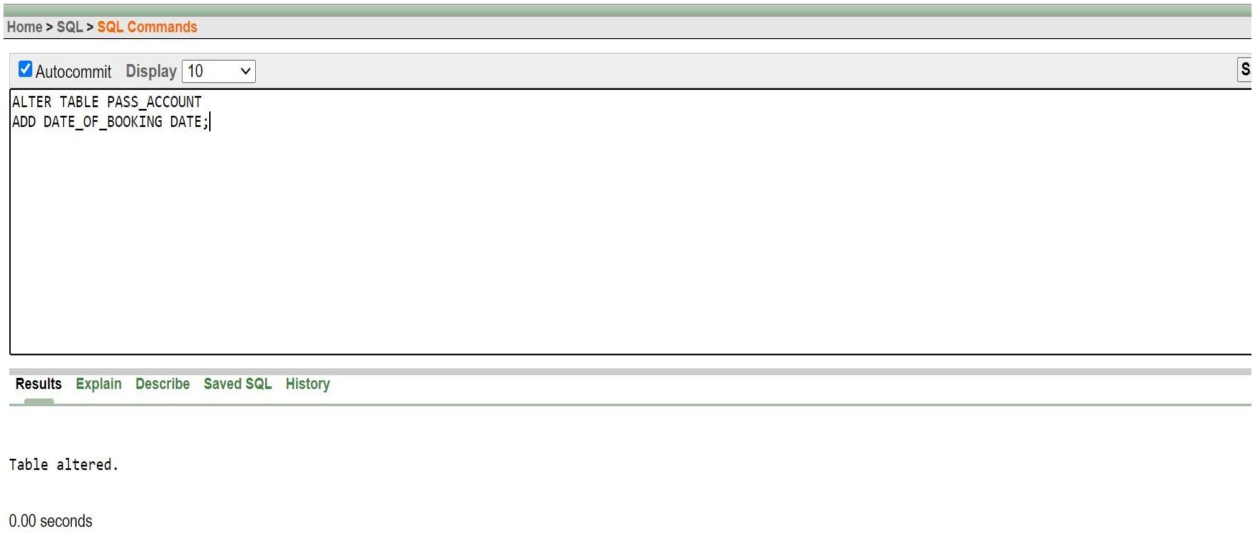
```
CREATE TABLE SEAT_AVAILABILITY
(
TR_NUM INT NOT NULL,
DATE_RES DATE NOT NULL,
CLASS VARCHAR(20) NOT NULL,
QUOTA VARCHAR(20) NOT NULL,
TOTAL_SEATS INT NOT NULL,
SEATS_AVAIL INT NOT NULL,
FOREIGN KEY(TR_NUM) REFERENCES TRAIN(TRAIN_NO)
)
```



```
Table created.

0.00 seconds
```

ALTER TABLE COMMANDS:

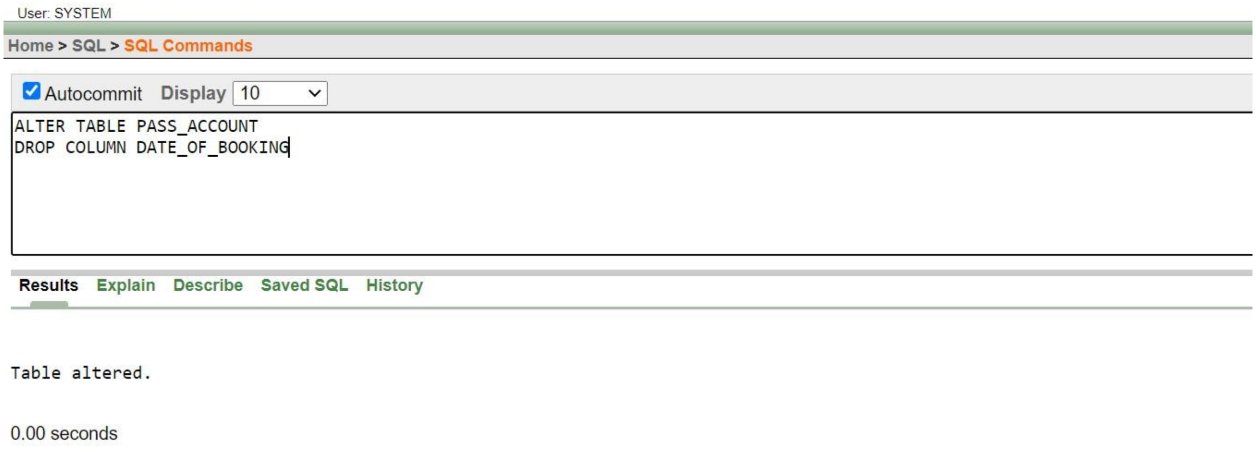


☑ Autocommit Display 10

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

```
Table altered.

0.00 seconds
```



☑ Autocommit Display 10

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

```
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.

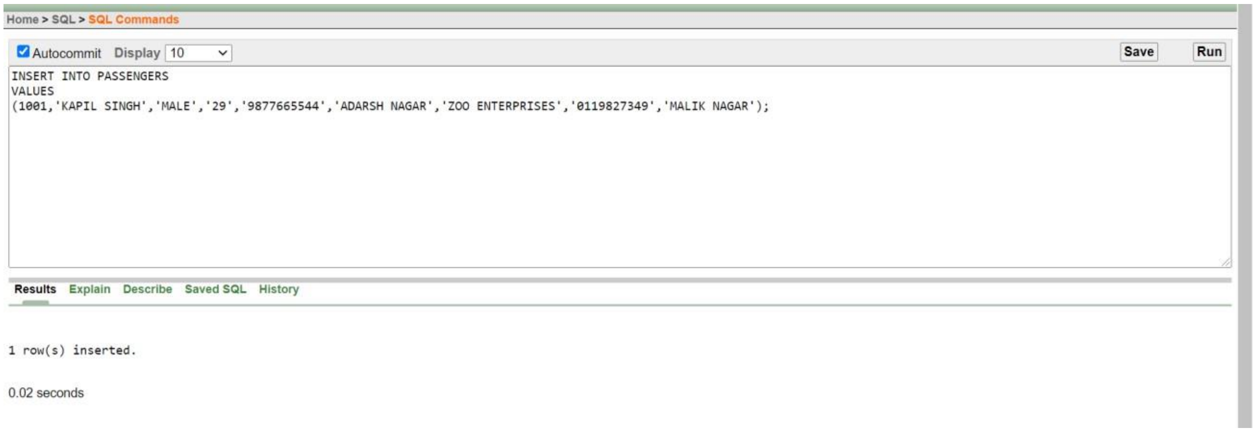
Syntax: DELETE FROM *table_name* WHERE *condition*;

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

Syntax: SELECT *column1*, *column2, ...* FROM *table_name*;

Implementing DML Commands:

1. **<u>INSERT:</u>**

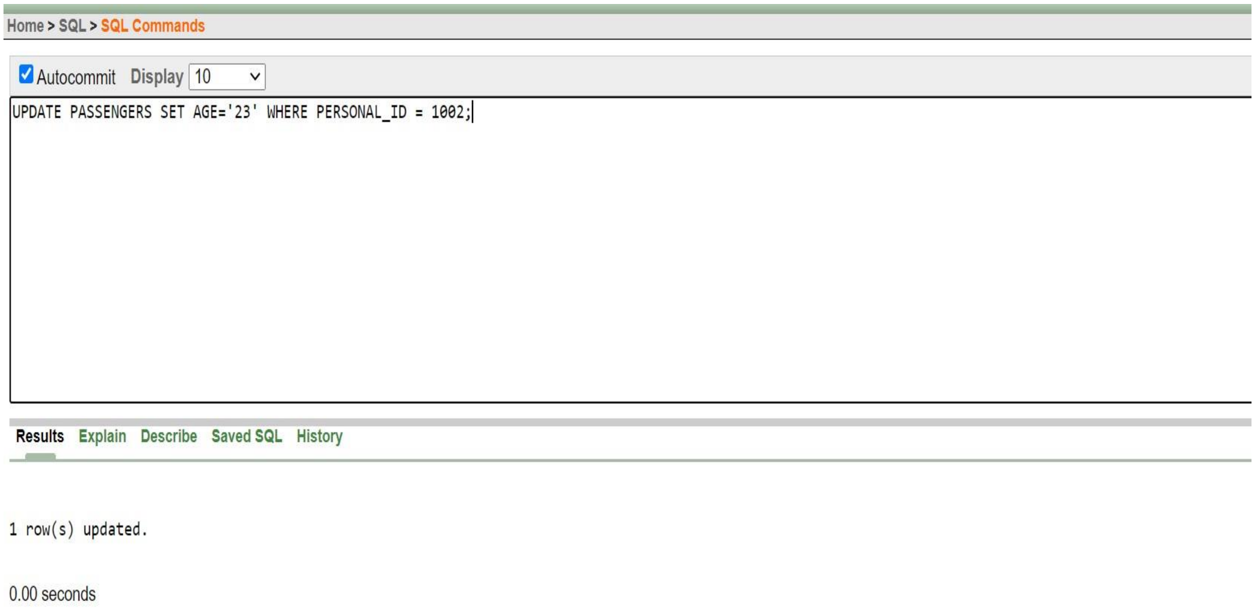

Home > SQL > SQL Commands
☑ Autocommit Display 10
Save
Run
Results Explain Describe Saved SQL History

```
INSERT INTO PASSENGERS
VALUES
(1001,'KAPIL SINGH','MALE','29','9877665544','ADARSH NAGAR','ZOO ENTERPRISES','0119827349','MALIK NAGAR');
```

```
1 row(s) inserted.

0.02 seconds
```

2. UPDATE:

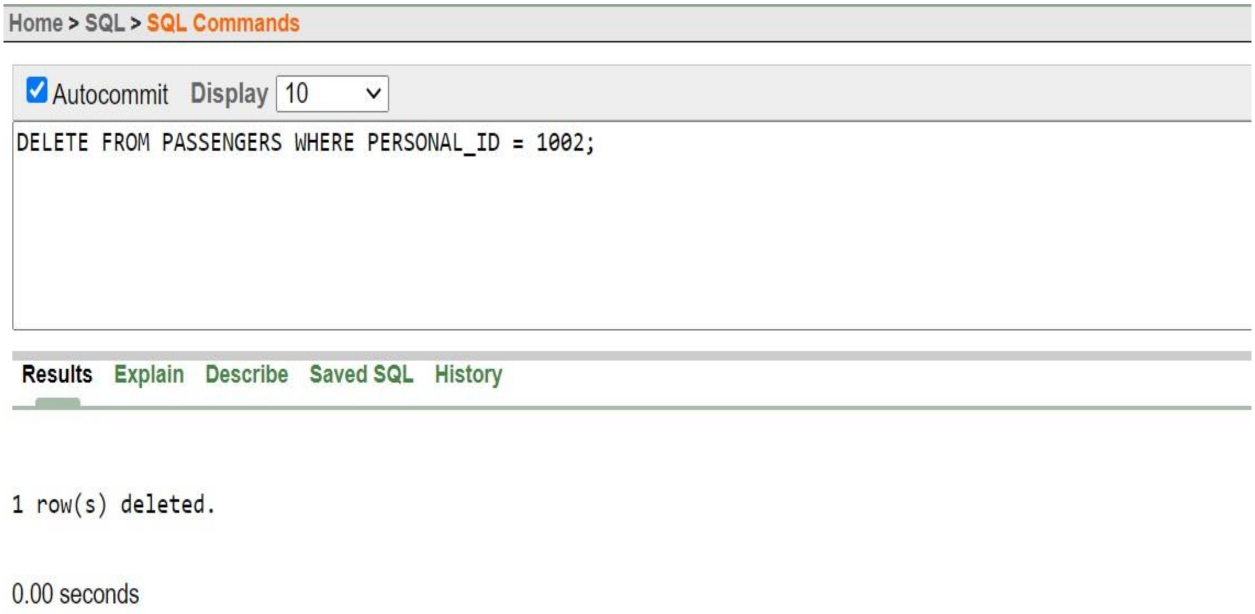

Home > SQL > SQL Commands
☑ Autocommit Display 10
Results Explain Describe Saved SQL History

```
UPDATE PASSENGERS SET AGE='23' WHERE PERSONAL_ID = 1002;
```

```
1 row(s) updated.

0.00 seconds
```

3. DELETE:

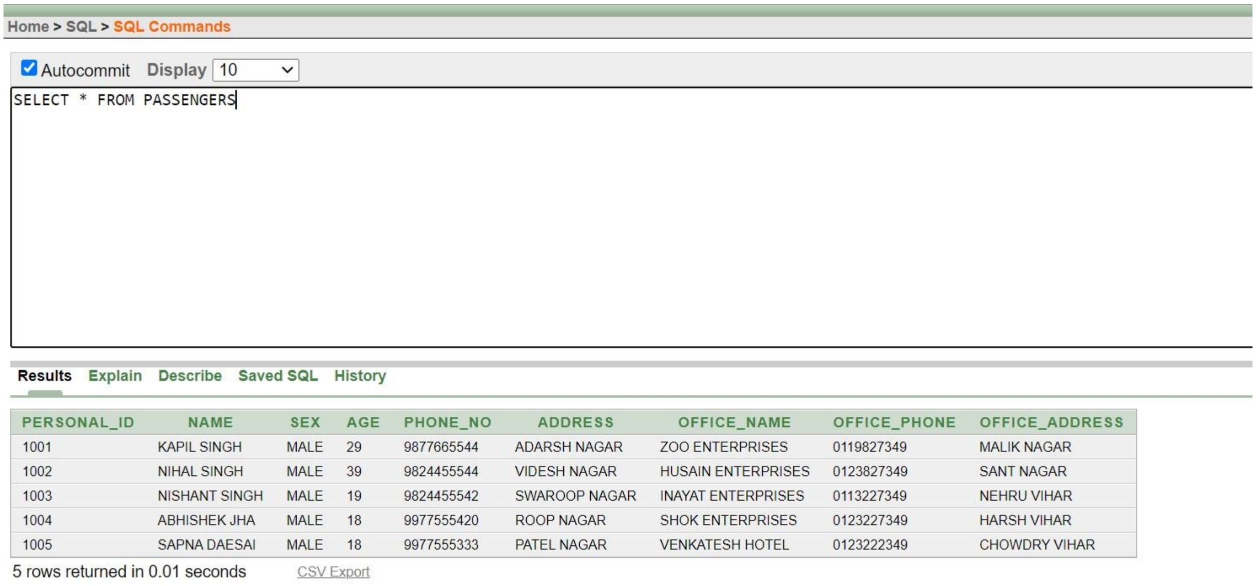


```
DELETE FROM PASSENGERS WHERE PERSONAL_ID = 1002;
```

```
1 row(s) deleted.
```

0.00 seconds

4. SELECT:



☑ Autocommit Display 10

```
SELECT * FROM PASSENGERS
```

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	CHOWDRY VIHAR

5 rows returned in 0.01 seconds

CONCLUSION:

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

EXPERIMENT 8

AIM: Performing Simple Queries.

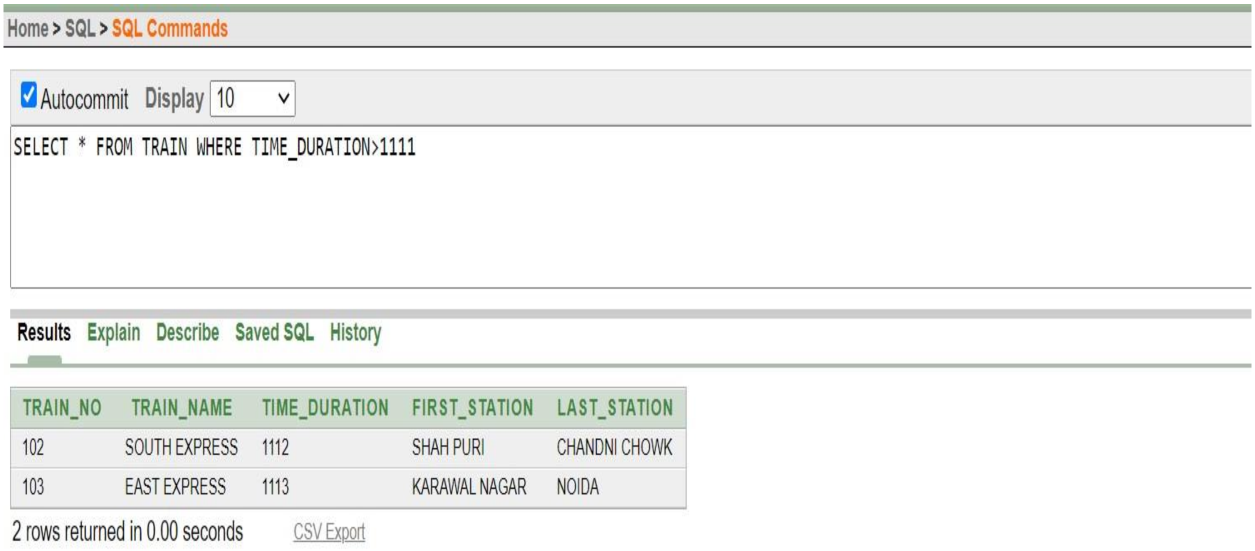
THEORY:

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.

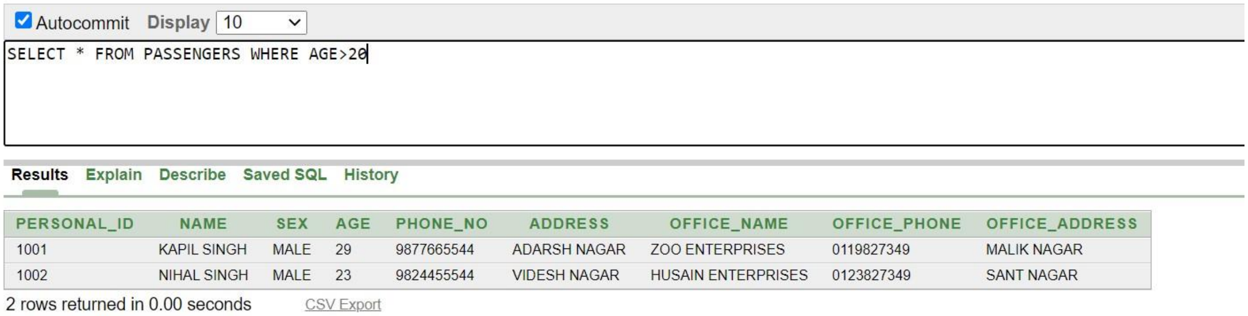


```
SELECT * FROM TRAIN WHERE TIME_DURATION>1111
```

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

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

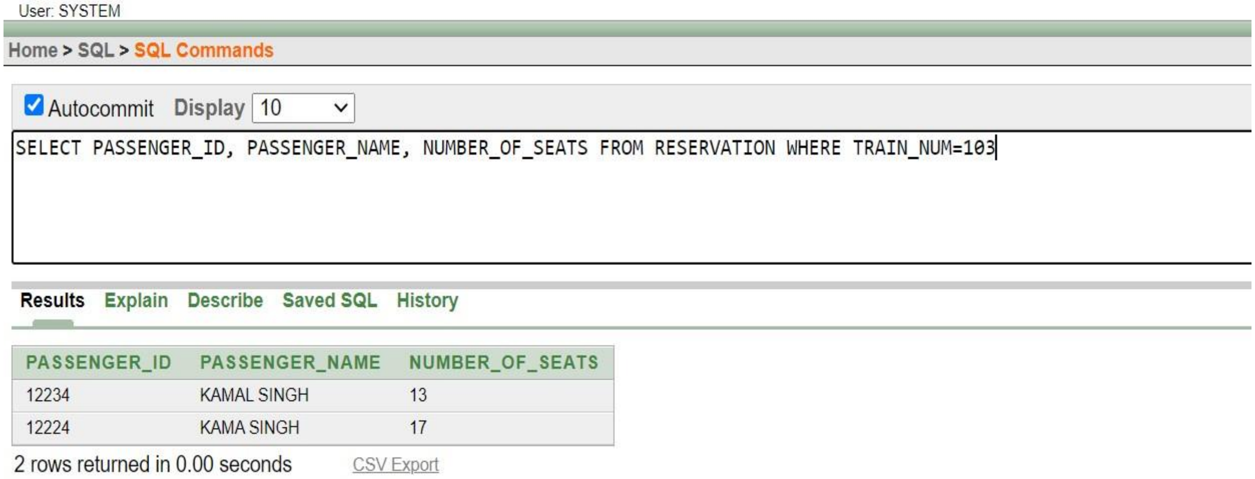


```
SELECT * FROM PASSENGERS WHERE AGE>20
```

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

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

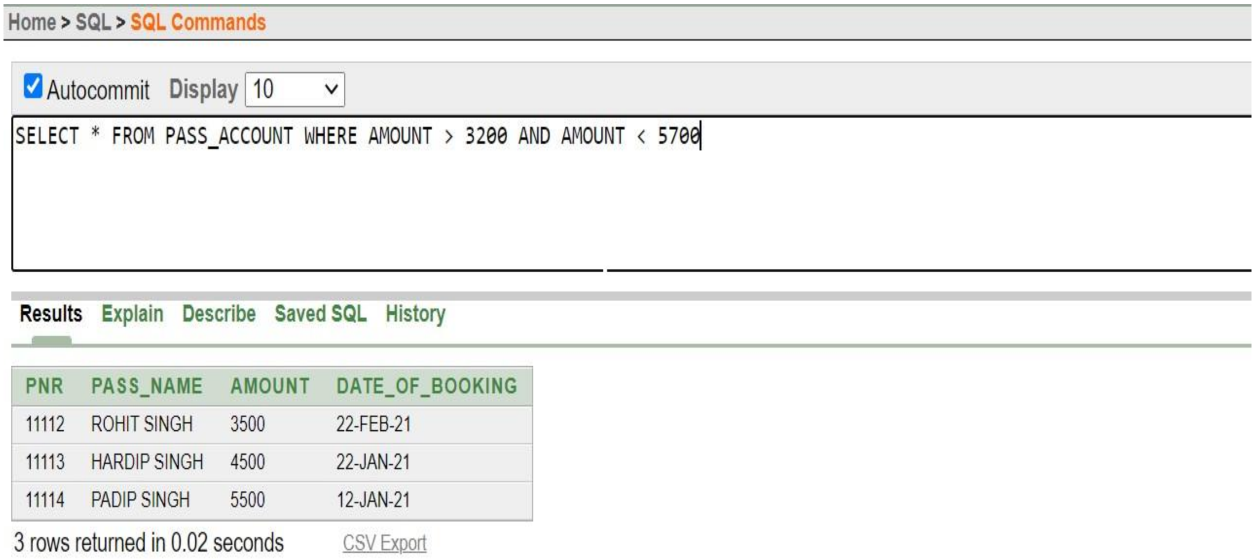


```
SELECT PASSENGER_ID, PASSENGER_NAME, NUMBER_OF_SEATS FROM RESERVATION WHERE TRAIN_NUM=103
```

PASSENGER_ID	PASSENGER_NAME	NUMBER_OF_SEATS
12234	KAMAL SINGH	13
12224	KAMA SINGH	17

2 rows returned in 0.00 seconds

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

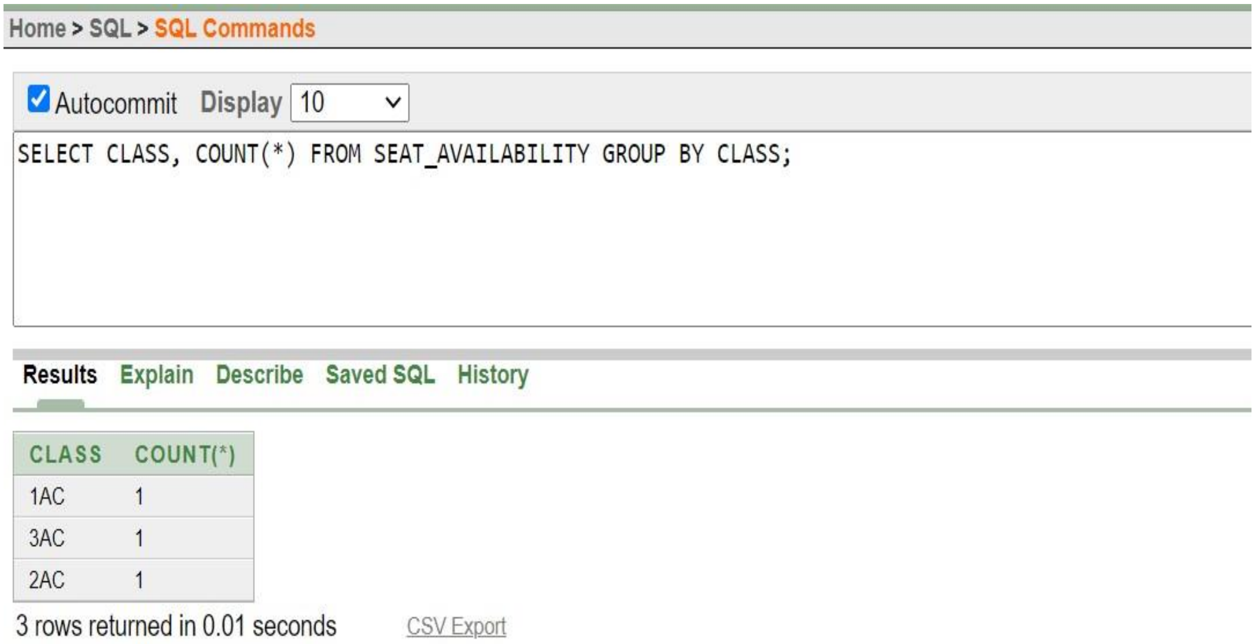


```
SELECT * FROM PASS_ACCOUNT WHERE AMOUNT > 3200 AND AMOUNT < 5700
```

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

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

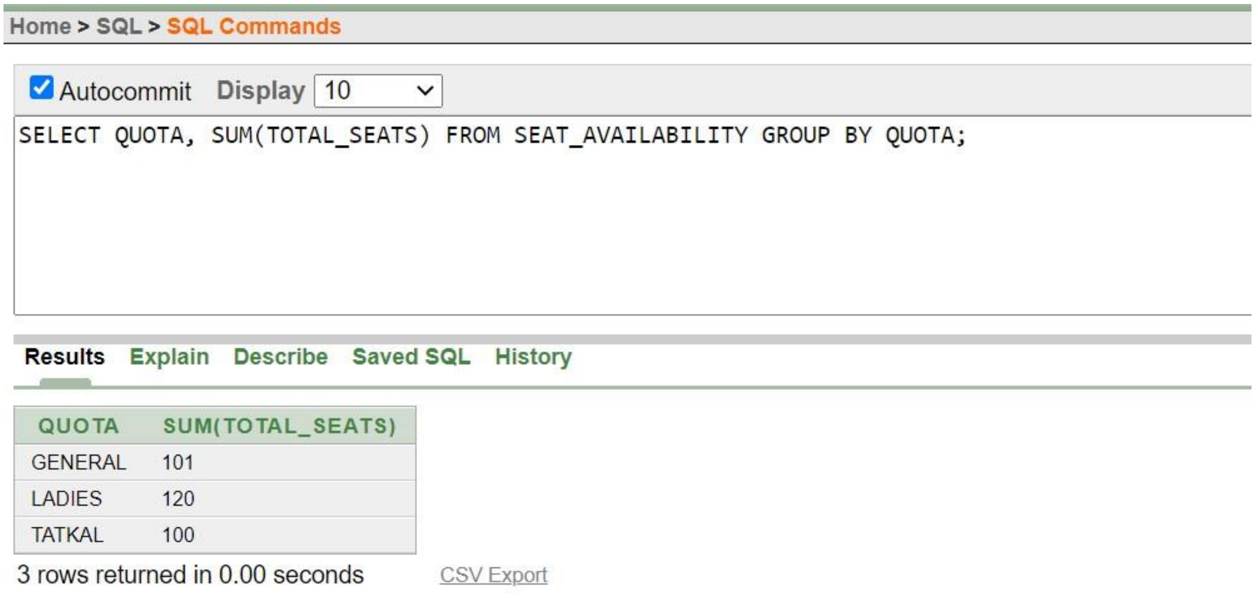


```
SELECT CLASS, COUNT(*) FROM SEAT_AVAILABILITY GROUP BY CLASS;
```

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

3 rows returned in 0.01 seconds

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

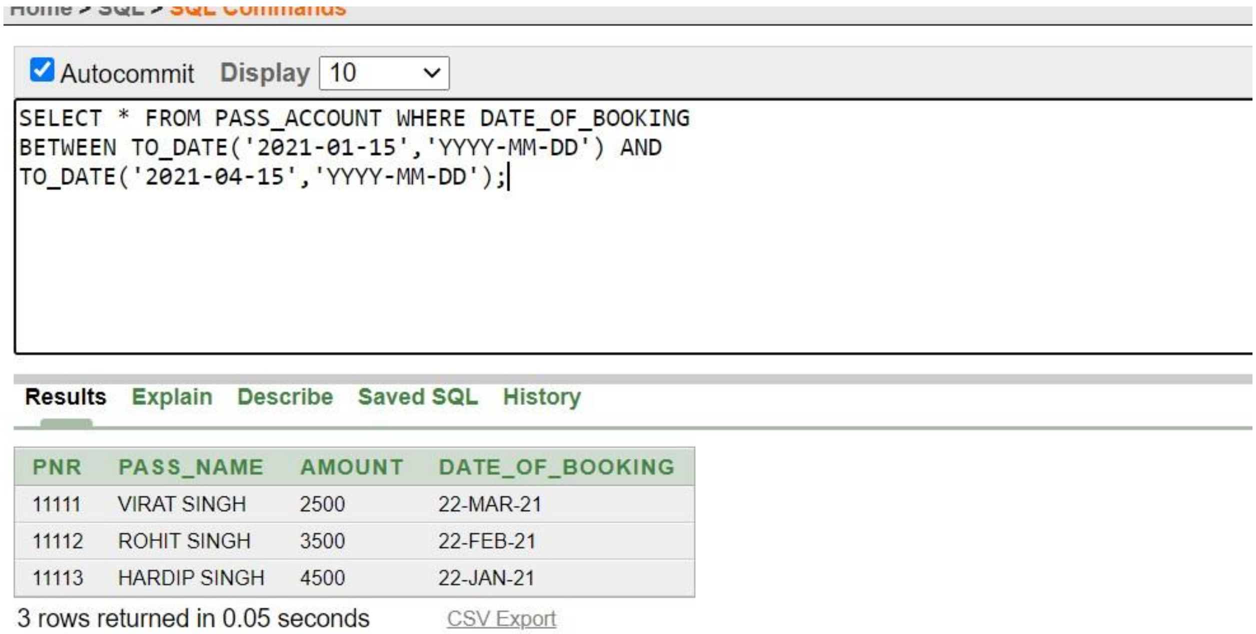


```
SELECT QUOTA, SUM(TOTAL_SEATS) FROM SEAT_AVAILABILITY GROUP BY QUOTA;
```

QUOTA	SUM(TOTAL_SEATS)
GENERAL	101
LADIES	120
TATKAL	100

3 rows returned in 0.00 seconds

7. **Retrieve information of all bookings from Account table made between 202101-15 and 2021-04-15.**



```
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');
```

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

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.

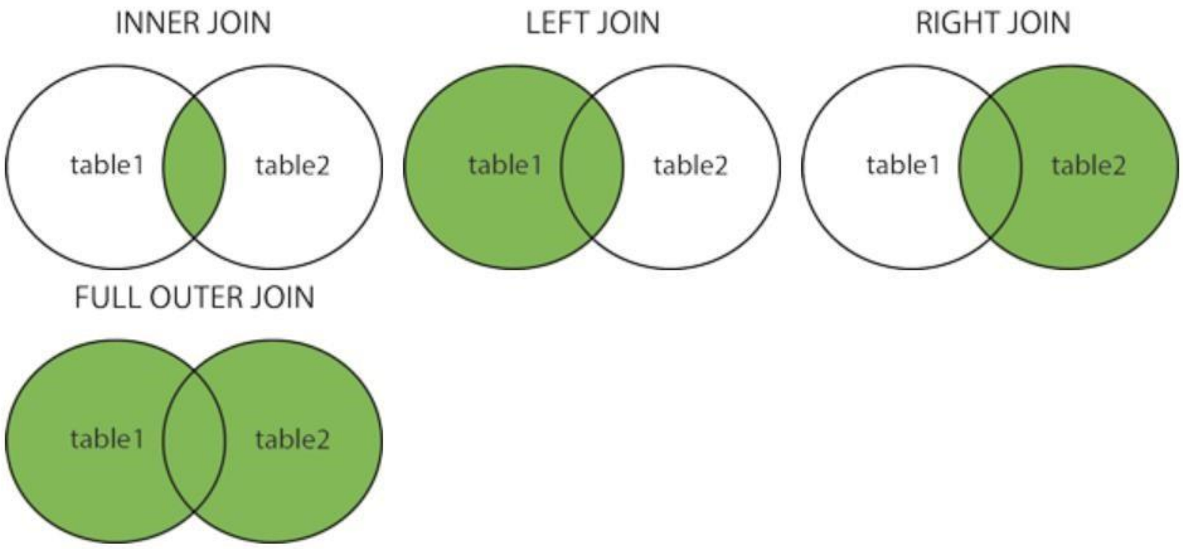
EXPERIMENT 9

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.

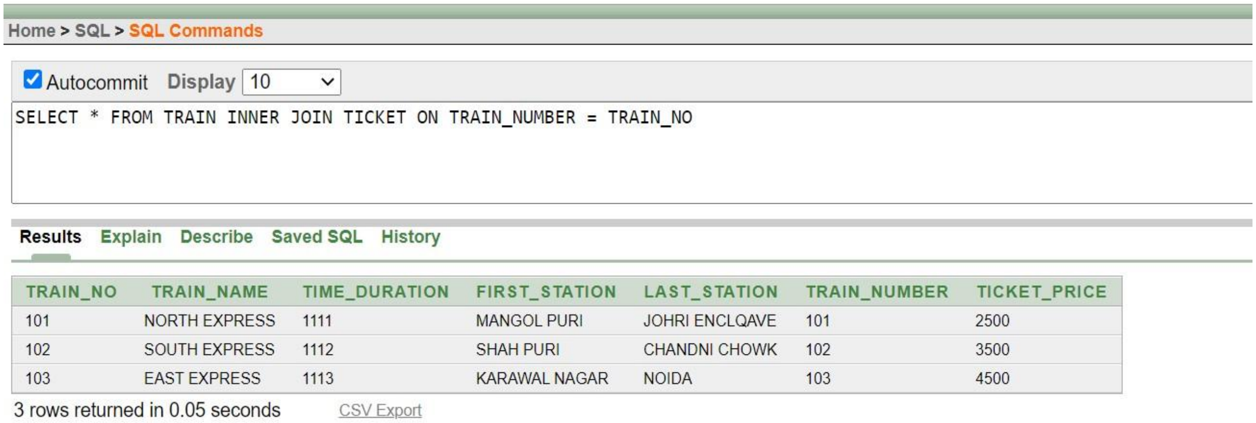
- **INNER JOIN:** This join is used to retrieve records that have matching values in both relations.
- **LEFT OUTER JOIN:** This join returns all records from the left relation and matching records from the right relation.
- **RIGHT OUTER JOIN:** This join returns all records from the right relation 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.clumn=table2.column WHERE Condition



☑ Autocommit Display 10

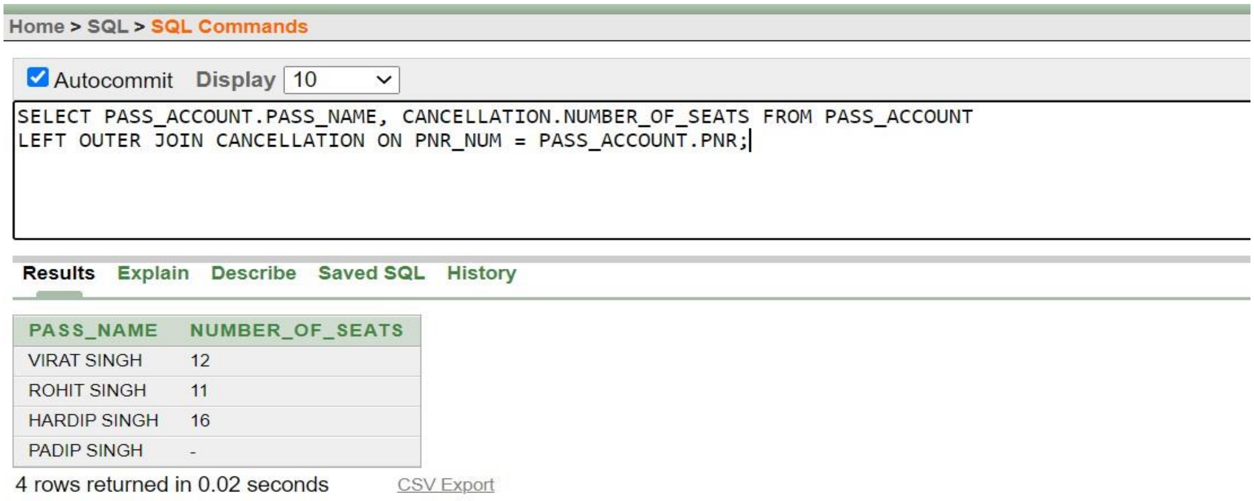
```
SELECT * FROM TRAIN INNER JOIN TICKET ON TRAIN_NUMBER = TRAIN_NO
```

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.clumn=table2.column WHERE Condition


Home > SQL > SQL Commands
☑ Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

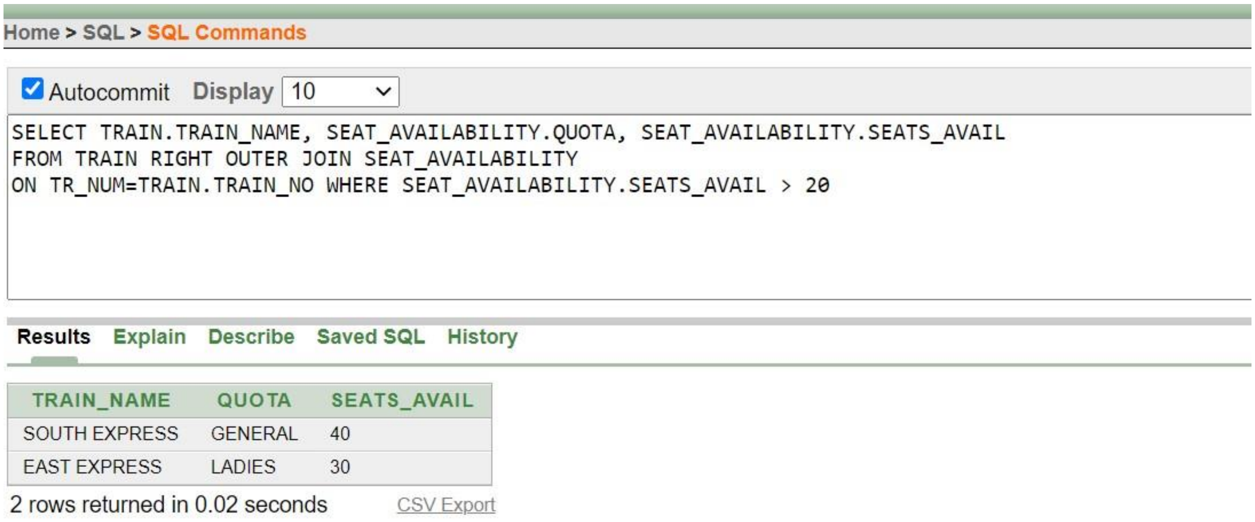
```
SELECT PASS_ACCOUNT.PASS_NAME, CANCELLATION.NUMBER_OF_SEATS FROM PASS_ACCOUNT
LEFT OUTER JOIN CANCELLATION ON PNR_NUM = PASS_ACCOUNT.PNR;
```

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

4 rows returned in 0.02 seconds

- **Right Outer Join**

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


Home > SQL > SQL Commands
☑ Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

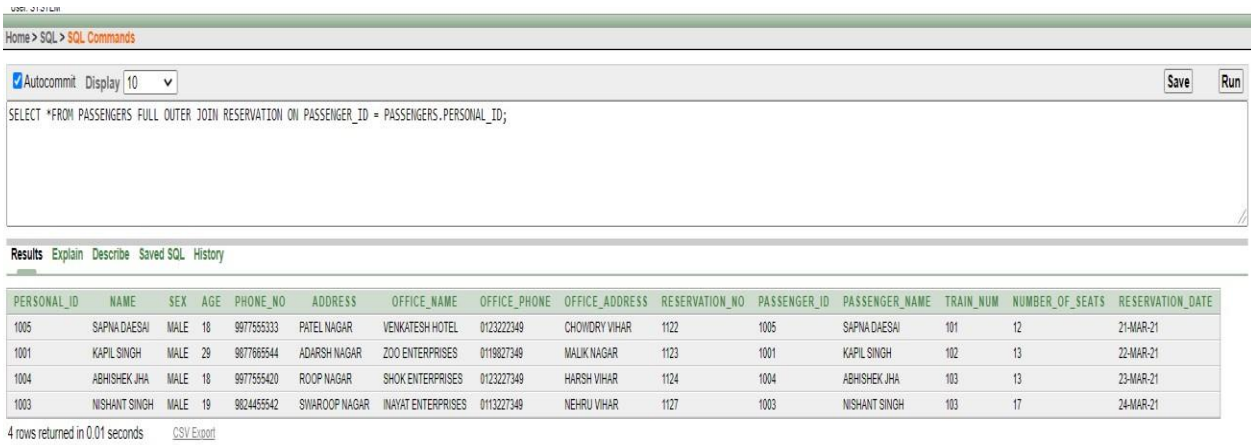
```
SELECT TRAIN.TRAIN_NAME, SEAT_AVAILABILITY.QUOTA, SEAT_AVAILABILITY.SEATS_AVAIL
FROM TRAIN RIGHT OUTER JOIN SEAT_AVAILABILITY
ON TR_NUM=TRAIN.TRAIN_NO WHERE SEAT_AVAILABILITY.SEATS_AVAIL > 20
```

TRAIN_NAME	QUOTA	SEATS_AVAIL
SOUTH EXPRESS	GENERAL	40
EAST EXPRESS	LADIES	30

2 rows returned in 0.02 seconds

- **Full Outer Join**

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


Home > SQL > SQL Commands
☑ Autocommit Display 10
Save
Run
Results Explain Describe Saved SQL History
CSV Export

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

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	CHOWDRY VIHAR	1122	1005	SAPNA DAESAI	101	12	21-MAR-21
1001	KAPIL SINGH	MALE	29	9877665544	ADARSH NAGAR	ZOO ENTERPRISES	0119827349	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	9824455542	SWAROOP NAGAR	INAYAT ENTERPRISES	0113227349	NEHRU VIHAR	1127	1003	NISHANT SINGH	103	17	24-MAR-21

4 rows returned in 0.01 seconds

<u>Conclusion</u>: We understood how to perform various types of joins in a database and how they work.

EXPERIMENT NO – 10

AIM: Implementing Aggregate functions, Date functions, Time functions.

THEORY: Aggregate functions return a single value based on groups of rows, rather than single value for each row. You can use AGGREGATE function in select lists and in ORDER BY and HAVING clauses. They are commonly used with the GROUP BY clause in a SELECT statement, where Oracle divides the rows of a queried table or view into groups.

The important Aggregate functions are:

- Avg
- Sum
- Max
- Min
- Count

❖ AVG

Returns the average value of

expression. Example:

The following query returns the average salary of all the employees.
SELECT AVG(SAL) "Average Salary" FROM EMP;

```
Average Salary
---------------------------------


2400.40
```

❖ SUM

Returns the sum value of

expression. Example:

The following query returns the sum salary of all the employees. **SELECT SUM(SAL) “Total Salary” FROM EMP;**

```
Total Salary
 ---------------------------------


26500
```

- **MAX**

Returns the maximum value of

expression. Example:

The following query returns the maximum salary from all the employees. **SELECT MAX(SAL) “Max Salary” FROM EMP;**

```
Maximum Salary
 ---------------------------------

          4500
```

- **MIN**

Returns the minimum value of

expression. Example:

The following query returns the minimum salary from all the employees. **SELECT MIN(SAL) “Min Salary” FROM EMP;**

```
Minimum Salary
 ---------------------------------

1200
```

COUNT

Returns the number of rows in the query. If you specify expression then count ignore nulls. If you specify the asterisk(*) , this function returns all rows, including duplicates and nulls. COUNT never returns null.

Example:

The following query returns the number of employees. **SELECT COUNT FROM EMP;**

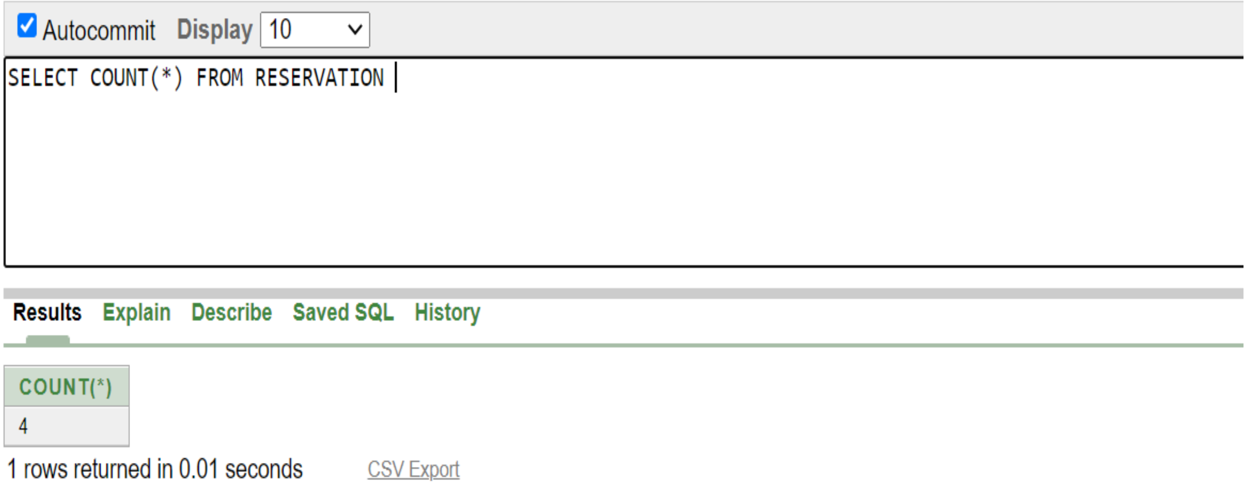
```
COUNT
 ---------------------------------
14
```

The following query counts the number of employees whose salary is not null. **SELECT COUNT(SAL) FROM EMP;**

```
COUNT
 ---------------------------------
12
```

AGGREGATE FUNCTIONS:

- COUNT()

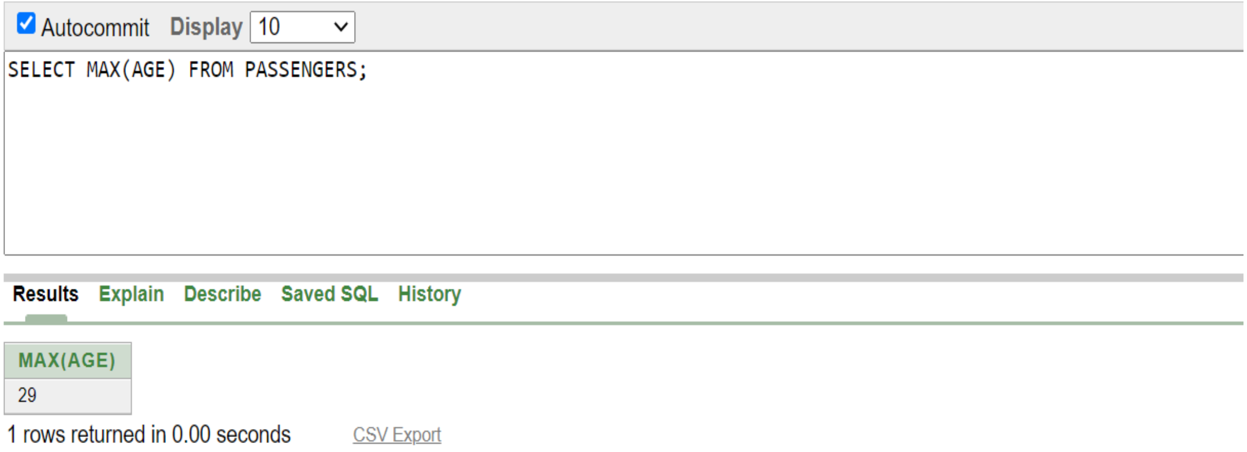


```
SELECT COUNT(*) FROM RESERVATION
```

```
COUNT(*)
4
```

1 rows returned in 0.01 seconds

- MAX()

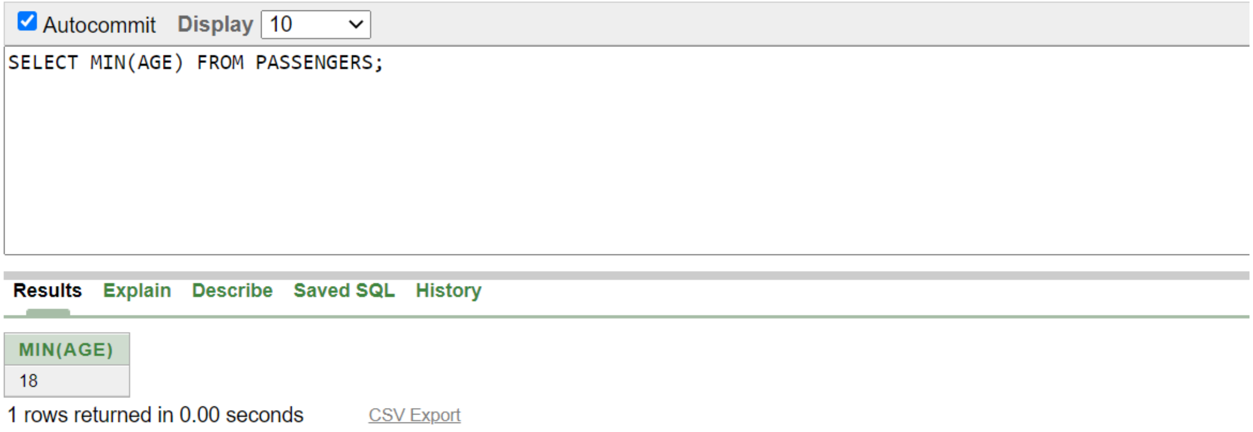


```
SELECT MAX(AGE) FROM PASSENGERS;
```

```
MAX(AGE)
29
```

1 rows returned in 0.00 seconds

- MIN()

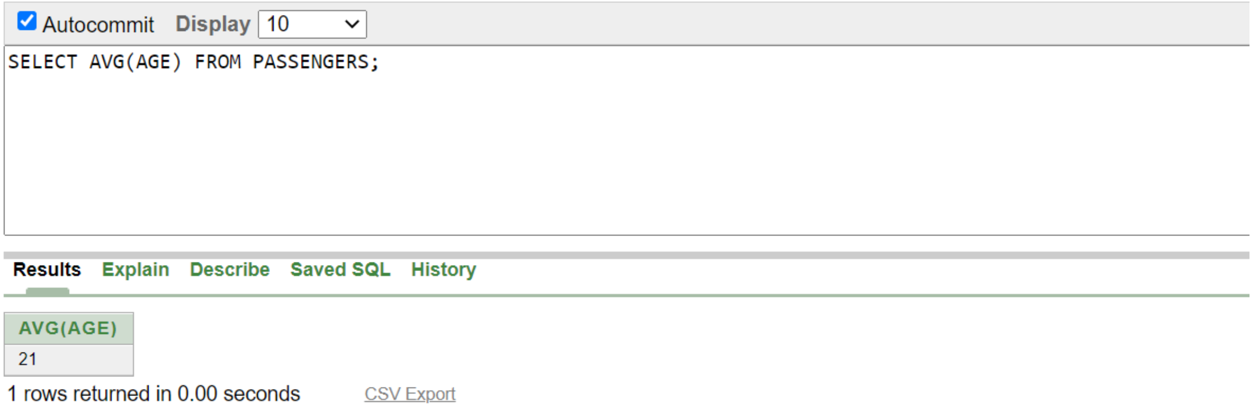


```
SELECT MIN(AGE) FROM PASSENGERS;
```

```
MIN(AGE)
18
```

1 rows returned in 0.00 seconds

- AVG()

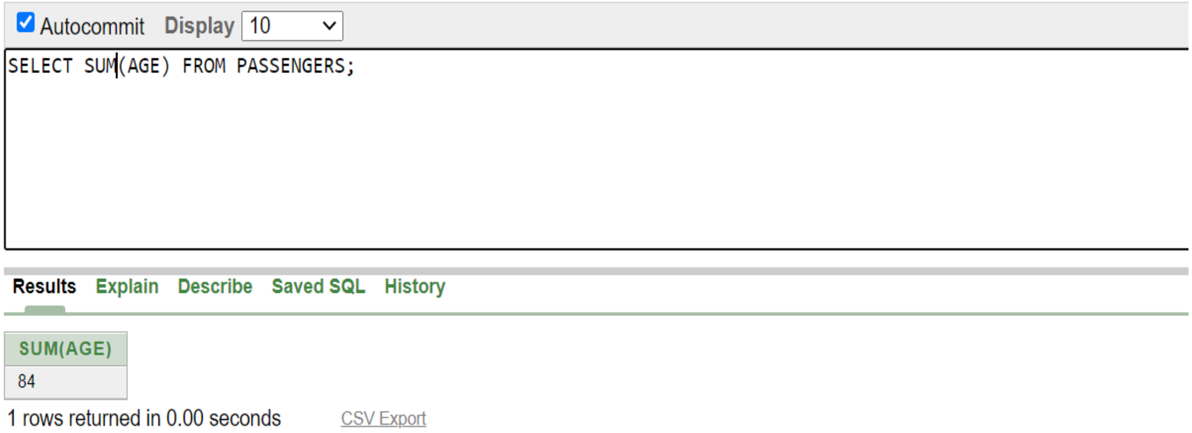


```
SELECT AVG(AGE) FROM PASSENGERS;
```

AVG(AGE)
21

1 rows returned in 0.00 seconds

- SUM()



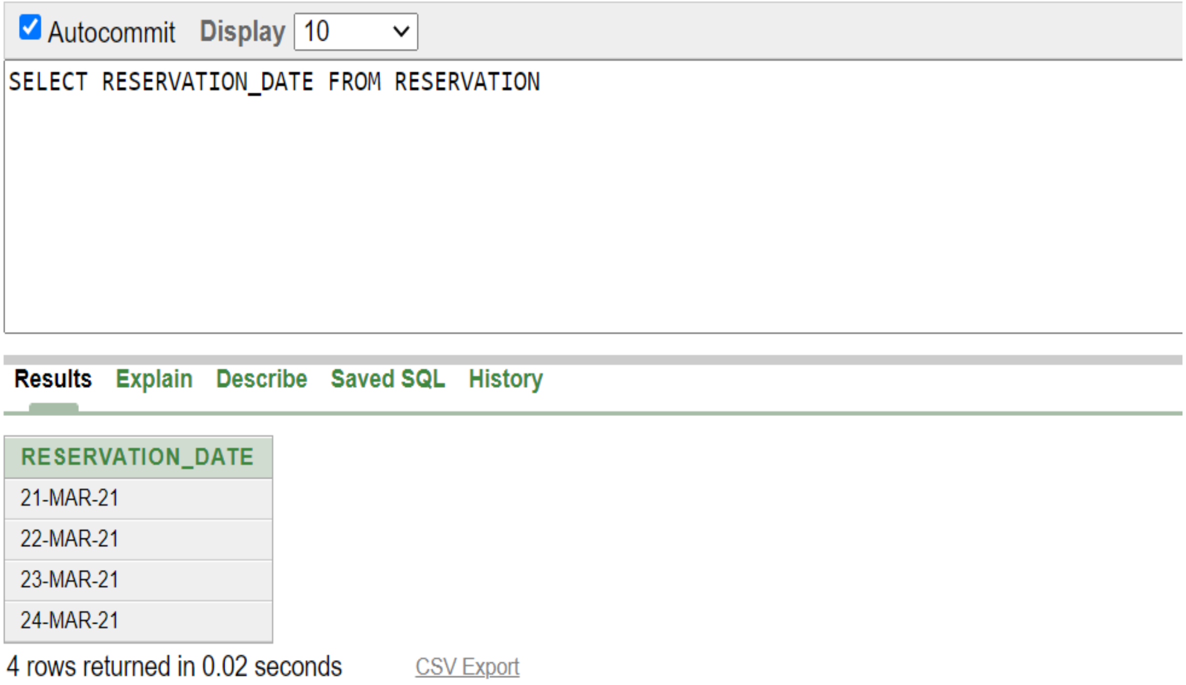
```
SELECT SUM(AGE) FROM PASSENGERS;
```

SUM(AGE)
84

1 rows returned in 0.00 seconds

IMPLEMENTING DATE AND TIME FUNCTIONS:

- Extract current date of the system.



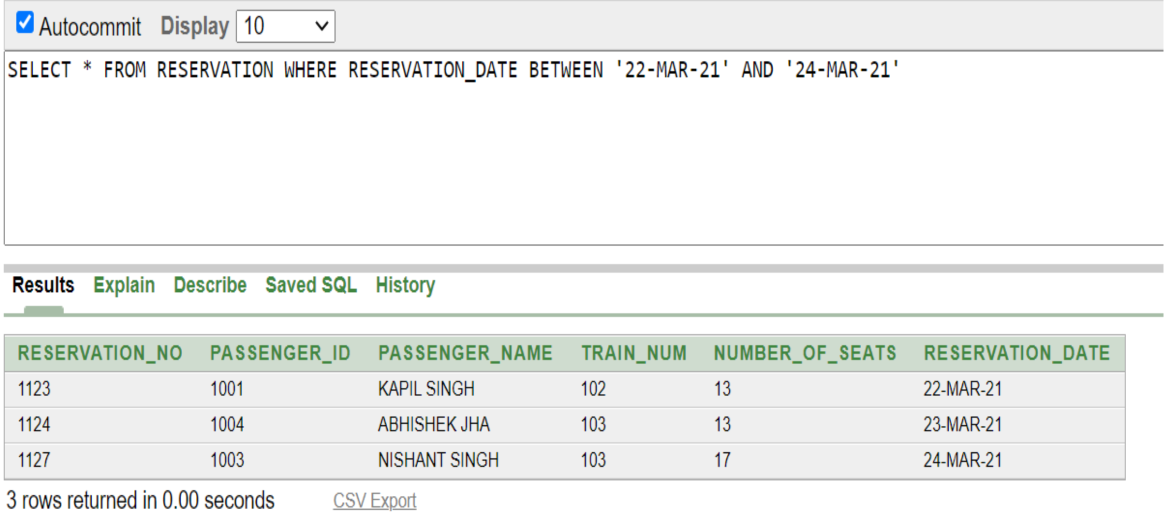
☑ Autocommit Display 10

```
SELECT RESERVATION_DATE FROM RESERVATION
```

RESERVATION_DATE
21-MAR-21
22-MAR-21
23-MAR-21
24-MAR-21

4 rows returned in 0.02 seconds

- Retrieve records of all the bookings made between March 22, 2021 and March 24, 2021



☑ Autocommit Display 10

```
SELECT * FROM RESERVATION WHERE RESERVATION_DATE BETWEEN '22-MAR-21' AND '24-MAR-21'
```

RESERVATION_NO	PASSENGER_ID	PASSENGER_NAME	TRAIN_NUM	NUMBER_OF_SEATS	RESERVATION_DATE
1123	1001	KAPIL SINGH	102	13	22-MAR-21
1124	1004	ABHISHEK JHA	103	13	23-MAR-21
1127	1003	NISHANT SINGH	103	17	24-MAR-21

3 rows returned in 0.00 seconds

CONCLUSION:

In this experiment, we were able to apply aggregate functions to various relations in the database which helped in better retrieval of valuable information about the data. Additionally, we also implemented date and time functions which helped us work with date and time attributes.

EXPERIMENT – 11

AIM: To implement Grant and Revoke statements, Views and the Triggers on Railway Management System.

THEORY:

1. Triggers

A trigger is a special kind of stored procedure that automatically executes when an event occurs in the Database Server. DML triggers execute when a user tries to modify data through a data manipulation language (DML) event. DML events are INSERT, UPDATE or DELETE statements on a table or view.

Benefits of Triggers

Following are the benefits of triggers:

- Generating some derived column values automatically.
- Enforcing referential integrity.
- Event logging and storing information on table access.
- Auditing.
- Synchronous replication of tables.
- Imposing security authorisations.
- Preventing invalid transactions.

Trigger Classification

Triggers can be classified based on the following parameters.

- Classification based on the timing

BEFORE Trigger: It fires before the specified event has occurred. AFTER Trigger: It fires after the specified event has occurred.
INSTEAD OF Trigger: A special Type.

- Classification based on the level

STATEMENT Level Trigger: It fires one time for the specified event statement. ROW Level Trigger: It fires for each record that got affected in the specified event.

- Classification based on the Event

DML Trigger: It fires when the DML event is specified (INSERT/UPDATE/DELETE). DDL Trigger: It fires when the DDL event is specified (CREATE/ALTER).
DATABASE Trigger: It fires when the database event is specified (LOGON/LOGOFF/STARTUP/SHUTDOWN).

So, each trigger is the combination of above parameters.

1. GRANT AND REVOKE STATEMENTS

DCL commands are used to enforce database security in a multiple user database environment. Two types of DCL commands are GRANT and REVOKE. Only Database Administrator or owner of the database object can provide/remove privileges on database object.

2. SQL GRANT

SQL GRANT is a command used to provide access or privileges on the database objects to the users.

SYNTAX

```
GRANT privilege_name

ON object_name

TO{user_name | PUBLIC | roel_name}

[WITH GRANT OPTION];
```

- **privilege_name** is the access right or privilege granted to the user. Some of the access rights are ALL, EXECUTE and SELECT.

- **object_name** is the name of an database object like TABLE, VIEW, STORED PROC and SEQUENCE.
  - **User_name** is the name of the user to whom an access right is being granted.
  - **PUBLIC** is used to grant access rights to all users.
  - **ROLES** are a set of privileges groups together.

3. SQL REVOKE

The REVOKE command removes user access rights or privileges to the database objects.

SYNTAX

```
REVOKE privilege_name

ON object_name

FROM {user_name | PUBLIC | role_name}
```

4. VIEWS

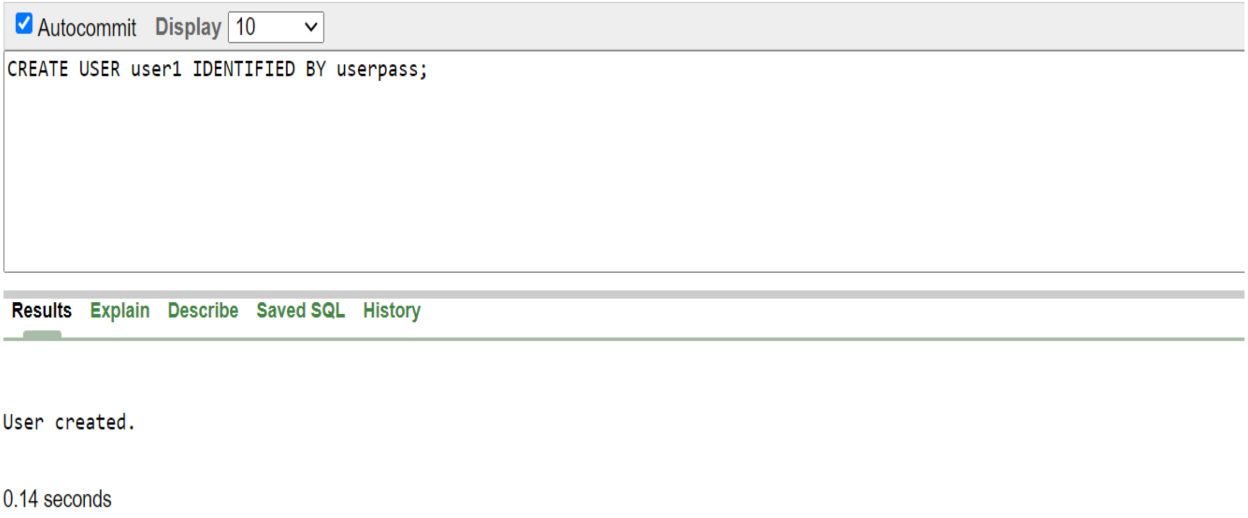
A VIEW is a virtual table, through which a selective portion of the data from one or more tables can be seen. Views do not contain data of their own. They are used to restrict access to the database or to hide complexity. A view is stored as a SELECT statement in the database. DML operations on a view like INSERT, UPDATE, DELETE affects the data in the original table upon which the view is based.

USES:

- Structure data in a way that users or classes of users find natural or intuitive.

GRANT AND REVOKE QUERIES

1. Creating User:

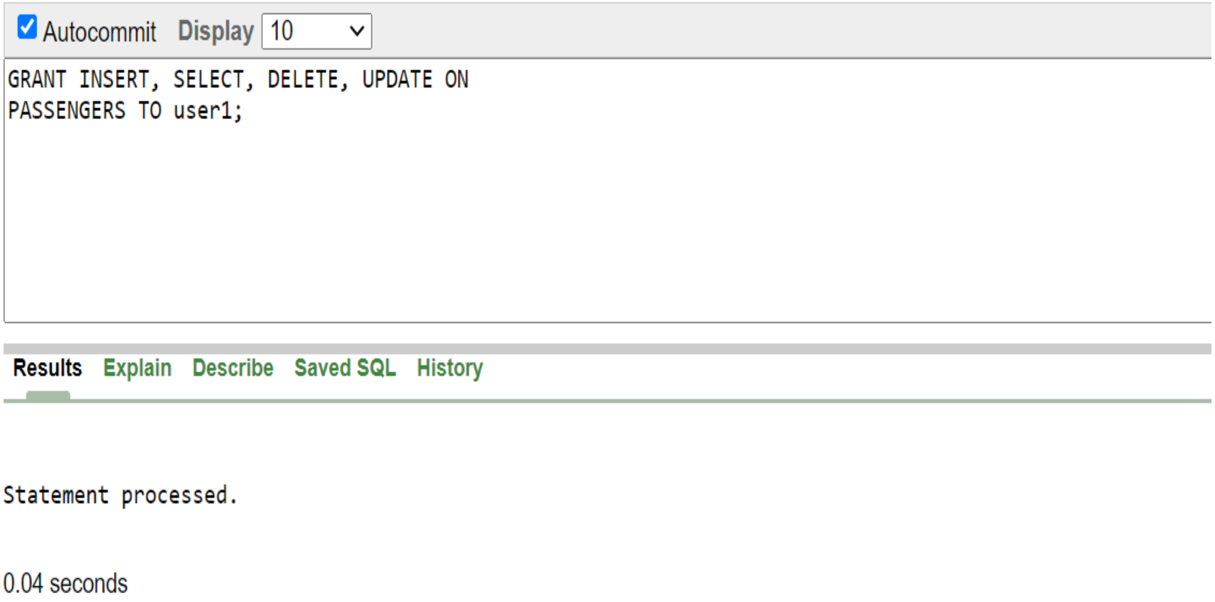


```
CREATE USER user1 IDENTIFIED BY userpass;
```

```
User created.
```

0.14 seconds

2. **Grant Insert, Select, Update, Delete on PASSENGERS to user1**

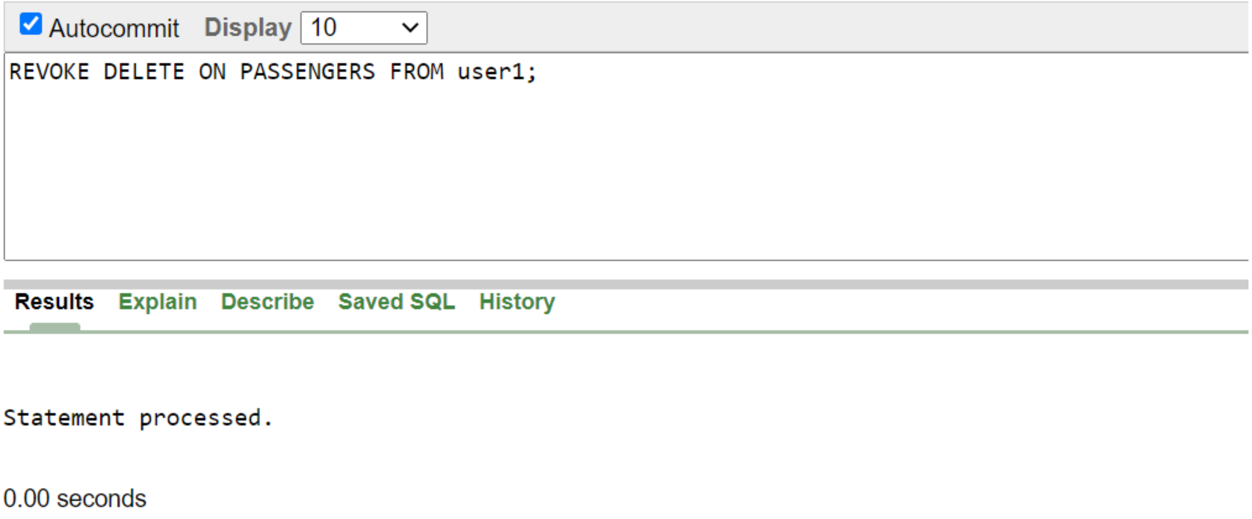


```
GRANT INSERT, SELECT, DELETE, UPDATE ON
PASSENGERS TO user1;
```

```
Statement processed.
```

0.04 seconds

3. Revoke delete



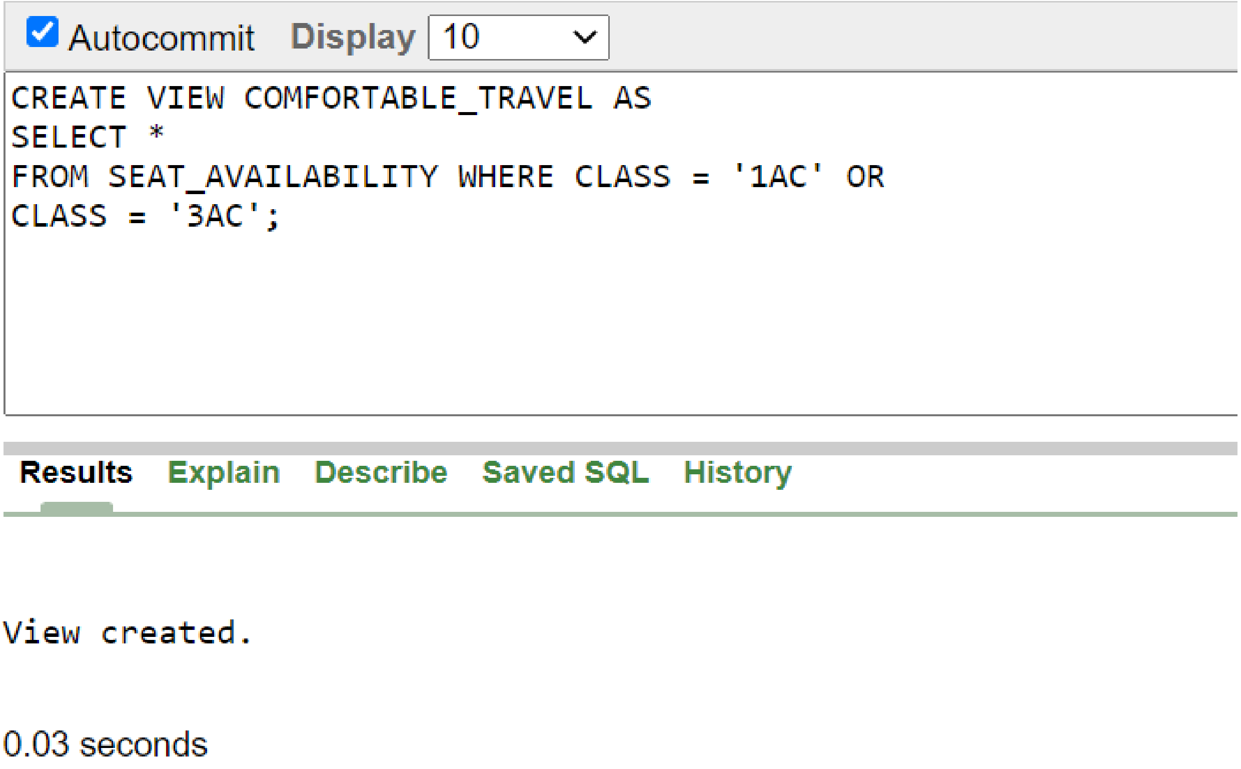
```
REVOKE DELETE ON PASSENGERS FROM user1;
```

```
Statement processed.
```

0.00 seconds

VIEWS

1. CREATE VIEW COMFORTABLE_TRAVEL

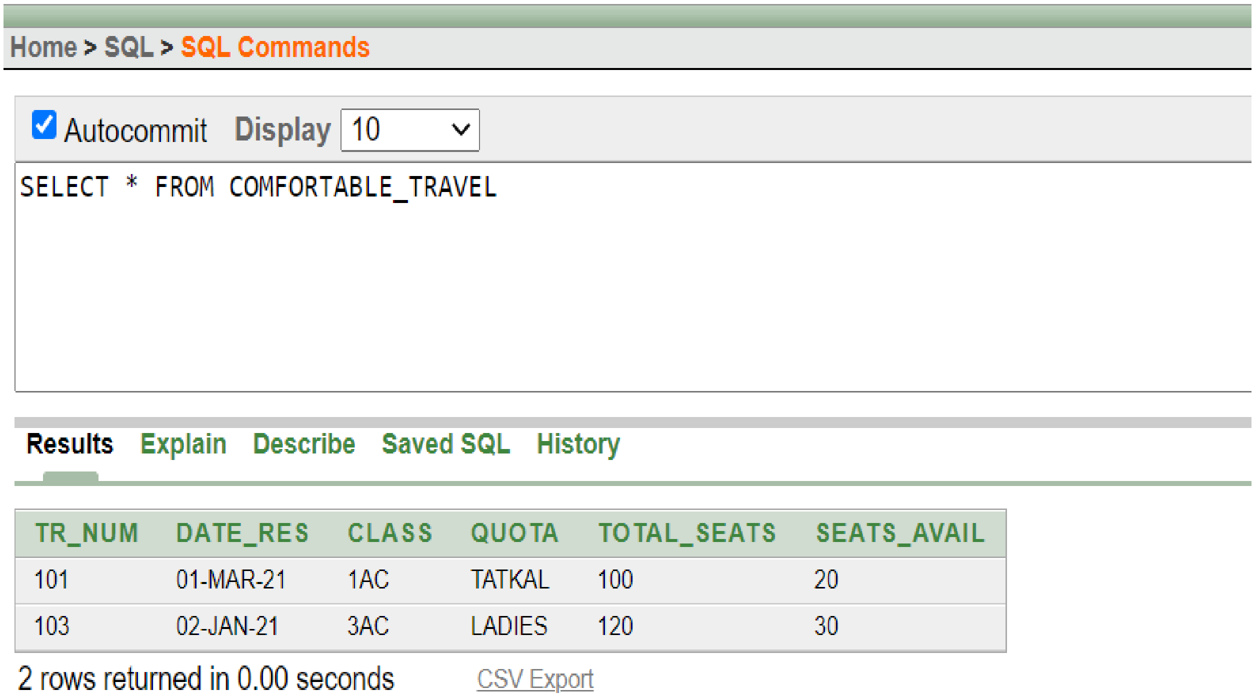


```
CREATE VIEW COMFORTABLE_TRAVEL AS
SELECT *
FROM SEAT_AVAILABILITY WHERE CLASS = '1AC' OR
CLASS = '3AC';
```

```
View created.
```

0.03 seconds

2. SELECT STATEMENT O COMFORTABLE_VIEW



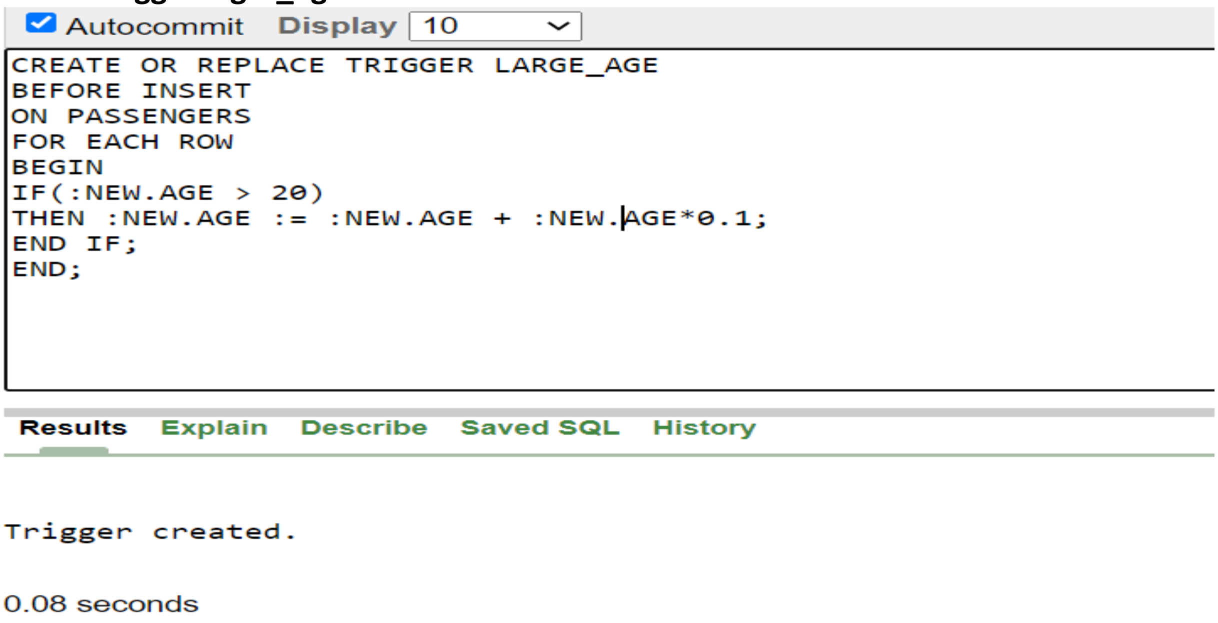
```
SELECT * FROM COMFORTABLE_TRAVEL
```

TR_NUM	DATE_RES	CLASS	QUOTA	TOTAL_SEATS	SEATS_AVAIL
101	01-MAR-21	1AC	TATKAL	100	20
103	02-JAN-21	3AC	LADIES	120	30

2 rows returned in 0.00 seconds

TRIGGERS

1. Create trigger lager_age

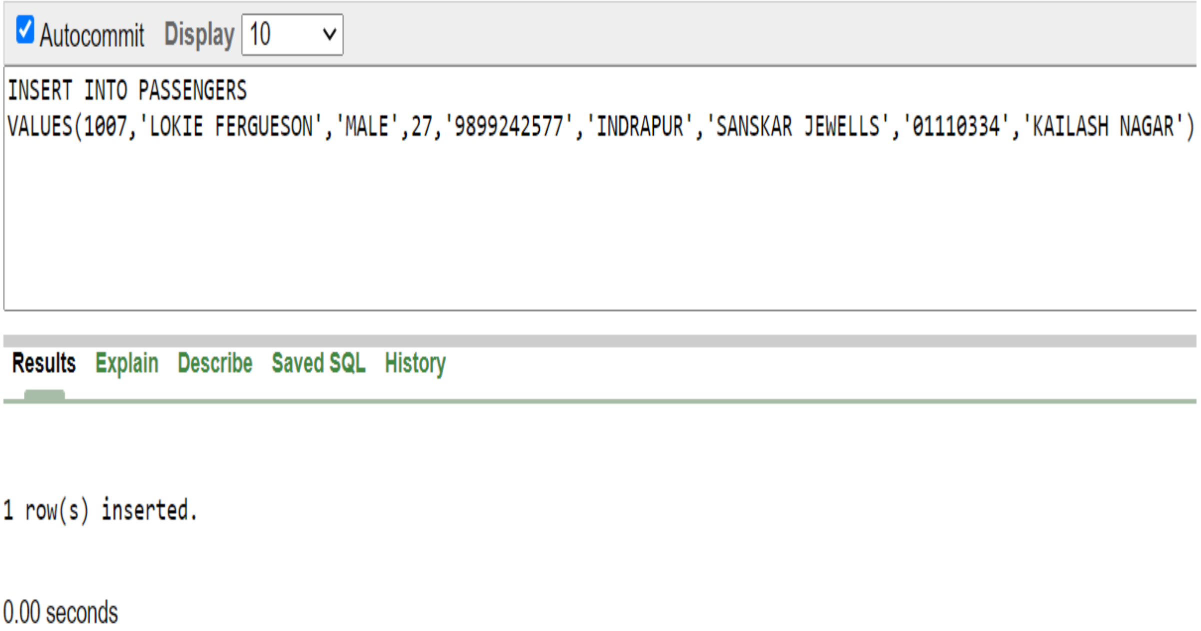


```
CREATE OR REPLACE TRIGGER LARGE_AGE
BEFORE INSERT
ON PASSENGERS
FOR EACH ROW
BEGIN
IF(:NEW.AGE > 20)
THEN :NEW.AGE := :NEW.AGE + :NEW.AGE*0.1;
END IF;
END;
```

```
Trigger created.

0.08 seconds
```

2. INSERT STATEMENT ON PASSENGERS

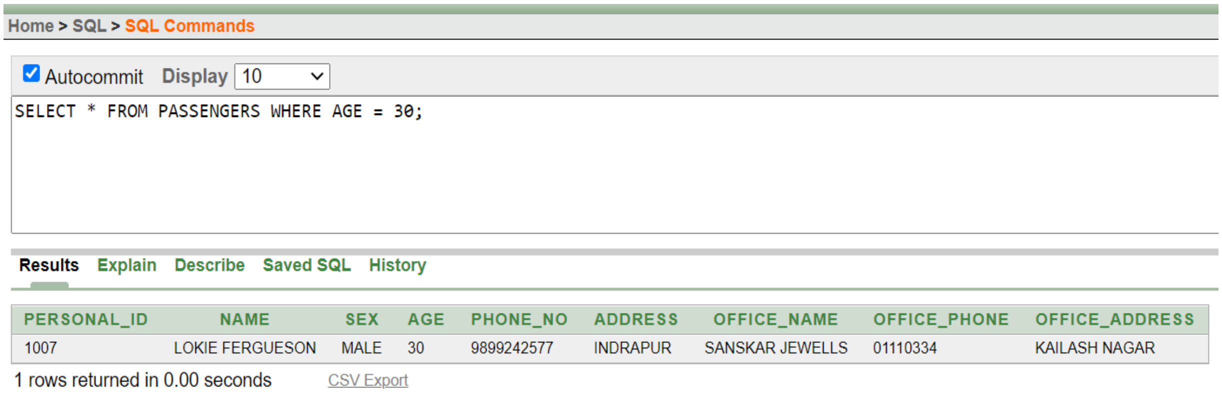


```
INSERT INTO PASSENGERS
VALUES(1007,'LOKIE FERGUESON','MALE',27,'9899242577','INDRAPUR','SANSKAR JEWELLS','01110334','KAILASH NAGAR')
```

```
1 row(s) inserted.

0.00 seconds
```

3. IMPACT OF AGE AFTER TRIGGER ON INSERTION


Home > SQL > SQL Commands
☑ Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT * FROM PASSENGERS WHERE AGE = 30;
```

PERSONAL_ID	NAME	SEX	AGE	PHONE_NO	ADDRESS	OFFICE_NAME	OFFICE_PHONE	OFFICE_ADDRESS
1007	LOKIE FERGUESON	MALE	30	9899242577	INDRAPUR	SANSKAR JEWELLS	01110334	KAILASH NAGAR

1 rows returned in 0.00 seconds

CONCLUSION:

In this experiment, we learnt how the significance of triggers and also implemented them. The impact of the trigger was also observed by inserting a record that would cause the trigger to initiate. Additionally, we also implemented views and observed the selections made based on the criteria provided. Finally, grant and revoke commands were also implemented.