

Module 1: SQL Server

1. Perform the normalization

- Create tables as per normalization
- Insert the data
- Join the table
- Create different views

Scenario:

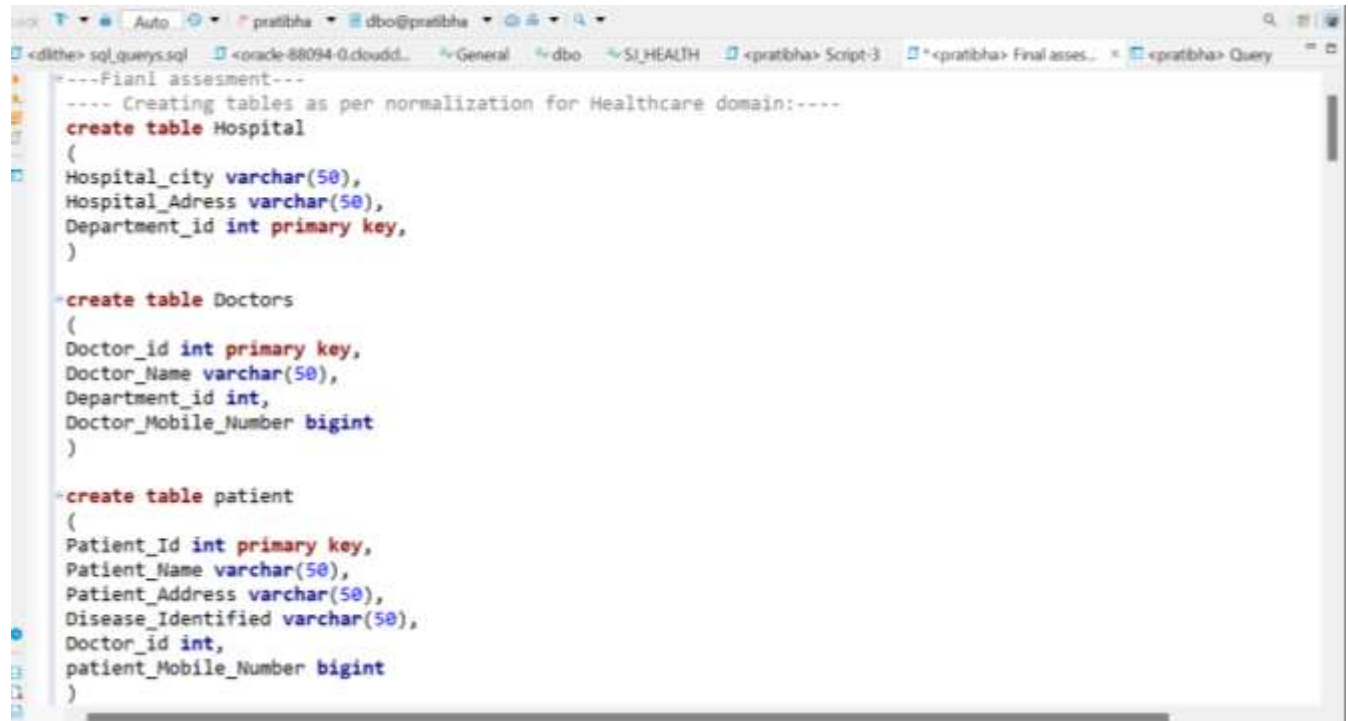
Healthcare domain:

Multiple patients visiting hospitals located in multiple cities, for check up and treatment, multiple doctors treating patients belong to specific department advising routine checkups(ex: Xray, Sugar test, urine test, MRI etc) and further drugs(medicines) for the disease identified.

Answer

Created three tables as per the normalization that is hospital, Docotor and patient.

Below snapshot shows the query for creatinf tables



```
----Fianl assesment---
---- Creating tables as per normalization for Healthcare domain:----
create table Hospital
(
  Hospital_city varchar(50),
  Hospital_Address varchar(50),
  Department_id int primary key,
)

create table Doctors
(
  Doctor_id int primary key,
  Doctor_Name varchar(50),
  Department_id int,
  Doctor_Mobile_Number bigint
)

create table patient
(
  Patient_Id int primary key,
  Patient_Name varchar(50),
  Patient_Address varchar(50),
  Disease_Identified varchar(50),
  Doctor_id int,
  patient_Mobile_Number bigint
)
```

After creating the table, inserted the data into the Hospital tables.

Below snapshot shows the inserting data in to the hospital tables

```

)
---inserting values to hospital tabale---
insert into Hospital values ('manglore','stateBank',101),
('manglore','mainroad',103),
('banglore','2nd cross',102),
('banglore','Rajajinagar',104),
('banglore','Ramangar',105),
('Udupi','kundapur',107),
('Baglkot','terdal',111),
('banglore','vijanagar',121),
('banglore','jaynagar',141),
('bijapur','ramnagar',151),
('Shimogga','Badravathi',171),
('manglore','moodbidre',131),
('Belgavi','Jayanagar',181),
('banglore','gandinagar',161),
('Hubballi','2nd cross',198),
('Bagalkot','jamkhandi',134),
('banglore','yashwanthpura',144)

-----adding foreign key reference---

```

Below snapshot shows adding foreign key reference to the existing table by using alter tabale command
For doctors and patient table

```

-----adding foreign key reference---
Alter table Doctors
add constraint FK_Department_id
foreign key(Department_id)
references Hospital

Alter table patient
add constraint FK_Doctor_id
foreign key(Doctor_id)
references Doctors

-----adding notnull constraint--
ALTER TABLE patient
alter column Patient_Name varchar(50) NOT NULL

```

Name	Value
Updated Rows	0
Query	-----adding notnull constraint-- ALTER TABLE patient alter column Patient_Name varchar(50) NOT NULL
Finish time	Thu Oct 06 22:15:27 IST 2022

Below snapshot is for the inserting data for patient table

The screenshot shows a SQL Editor window with a query to insert data into the 'patient' table. The query is as follows:

```
insert into patient values(491,'John','Mangalore','Dengue',333,783546596),
(401,'Maggi','Bangalore','Leprosy',301,64758493652),(402,'Julie','Moodbidri','Chikungunya',305,94758493652),
(403,'Kris','Mumbai','Malaria',302,29389921),(404,'Jia','Venur','Rabies',304,78654332),(405,'Mia','Chikm',
(406,'Sheela','Moodbidri','Trachoma',303,3456728),(407,'Ravi','Mangalore','Leprosy',307,56432467),(408,'
(409,'Mel','Venur','Trematodiasis',305,2678345),(410,'Manoj','Belthangady','Dengue',301,7485631),(411,'R',
(412,'Jorge','Mangalore','Rabies',306,563728),(413,'Lavanya','Ujire','Scabies',308,7890654),(414,'Vishal',
(415,'Manish','Belthangady','Snakebite',307,5623467),(416,'Rishab','Mangalore','Snakebite',307,789065),(
(418,'Raj','Ujire','Scabies',309,6754328),(419,'Anvira','Bangalore','Trematodiasis',305,785546),(420,'S',
(421,'Chandra','Bangalore','Dengue',306,26782925),(422,'Desmond','Pune','Trachoma',307,568292),(423,'Kir',
(424,'Laxmi','Pune','Chikungunya',307,2678854),(425,'Ashish','Chikmangalore','Malaria',307,678956),(426,
```

The bottom pane shows the 'Query' tab with the same insert statement and a 'Results' tab showing the execution status.

Performing join operation for Doctor and Patient tables

Below snapshot shows the result inner join option

The screenshot shows a SQL Editor window with an INNER JOIN query. The query is as follows:

```
---join operation---
---inner join operation to doctor and patient table---
select H.Patient_id ,H.Patient_Name ,H.Patient_Address,H.Disease_Identified,H.Doctor_id,
Doc.Doctor_name,Doc.Department_id
from patient as H
inner join Doctors as Doc
on H.Doctor_id=Doc.Doctor_id
```

The bottom pane shows the 'Results' tab with the following data:

Patient_id	Patient_Name	Patient_Address	Disease_Identified	Doctor_id	Doctor_name	Department_id
401	Maggi	Bangalore	Leprosy	301	ajaya	101
402	Julie	Moodbidri	Chikungunya	305	riti	111
403	Kris	Mumbai	Malaria	302	mahesh	104
404	Jia	Venur	Rabies	304	bhavana	107
405	Mia	Chikmangalore	Snakebite	307	bhavya	115
406	Sheela	Moodbidri	Trachoma	303	swathi	105
407	Ravi	Mangalore	Leprosy	307	bhavya	115
408	Malkova	Ujire	Scabies	304	bhavana	107
409	Mel	Venur	Trematodiasis	305	riti	111
410	Manoj	Belthangady	Dengue	301	ajaya	101
411	Randy	Pune	Chikungunya	303	swathi	105
412	Jorge	Mangalore	Rabies	306	poonja	121
413	Lavanya	Ujire	Scabies	308	sakshi	171
415	Manish	Belthangady	Snakebite	307	bhavya	115

Below snapshot shows the result Right join option

The screenshot shows a SQL query editor with the following query:

```

-----right join operation to doctor and patient table
select H.Patient_Id ,H.Patient_Name ,H.Patient_Address,H.Disease_Identified,H.Doctor_id,
Doc.Doctor_name,Doc.Department_id
from patient as H
right join Doctors as Doc
on H.Doctor_id=Doc.Doctor_id
  
```

The results window displays the following data:

Patient_Id	Patient_Name	Patient_Address	Disease_Identified	Doctor_id	Doctor_name	Department_id
412	Jorge	Mangalore	Rabies	306	pooja	121
421	Chanda	Bangalore	Dengue	306	pooja	121
405	Mia	Chikmagalore	Snakebite	307	bhavya	151
407	Ravi	Mangalore	Leprosy	307	bhavya	151
415	Manish	Bellthangady	Snakebite	307	bhavya	151
416	Bishab	Mangalore	Snakebite	307	bhavya	151
402	Demond	Pune	Trachoma	307	bhavya	151
424	Lami	Pune	Chikungunya	307	bhavya	151
425	Ashish	Chikmagalore	Malaria	307	bhavya	151
413	Lavanya	Ujire	Scabies	308	sakshi	171
428	Shashi	Bangalore	Trematodases	308	sakshi	171
418	Raj	Ujire	Scabies	308	neha	131
426	Mohan	Bellthangady	Rabies	308	neha	131
423	Kiran	Venur	Dengue	311	nit	111
401	Manoj	Mangalore	Dengue	311	nit	111
401	John	Mangalore	Dengue	313	manoj	101

Below snapshot shows the result Full join option

The screenshot shows a SQL query editor with the following query:

```

-----full join operation to doctor and patient table
select H.Patient_Id ,H.Patient_Name ,H.Patient_Address,H.Disease_Identified,H.Doctor_id,
Doc.Doctor_name,Doc.Department_id
from patient as H
full join Doctors as Doc
on H.Doctor_id=Doc.Doctor_id
  
```

The results window displays the following data:

Patient_Id	Patient_Name	Patient_Address	Disease_Identified	Doctor_id	Doctor_name	Department_id
403	Maggi	Bangalore	Leprosy	301	sujaya	103
402	Julie	Moodbichi	Chikungunya	305	niti	111
405	Kris	Mumbai	Malaria	302	mahesh	104
404	Ira	Venur	Rabies	304	baviana	107
405	Mia	Chikmagalore	Snakebite	307	bhavya	151
406	Sheela	Moodbichi	Trachoma	303	swathi	105
407	Ravi	Mangalore	Leprosy	307	bhavya	151
408	Malkova	Ujire	Scabies	304	baviana	107
409	Mel	Venur	Trematodases	305	niti	111
410	Manoj	Bellthangady	Dengue	301	sujaya	103
411	Randy	Pune	Chikungunya	303	swathi	105
412	Jorge	Mangalore	Rabies	306	pooja	121
413	Lavanya	Ujire	Scabies	308	sakshi	171

Below snapshot shows the result Cross join option

The screenshot shows a SQL Editor window with the following query:

```

-----cross join option to doctor and patient table
select H.Patient_Id ,H.Patient_Name ,H.Patient_Address,H.Disease_Identified,H.Doctor_id,
Doc.Doctor_name,Doc.Department_id
from patient as H
cross join Doctors as Doc
  
```

The results pane displays a table with 324 rows and 7 columns:

Patient_id	Patient_Name	Patient_Address	Disease_Identified	Doctor_id	Doctor_name	Department_id
309	412 Jorga	Mangalore	Rabies	306 manoj		101
310	413 Lavanya	Ujire	Scabies	308 manoj		101
311	415 Manish	Belthangady	Snakebite	307 manoj		101
312	416 Rohab	Mangalore	Snakebite	307 manoj		101
313	417 Rie	Mumbai	Dengue	301 manoj		101
314	418 Raj	Ujire	Scabies	309 manoj		101
315	419 Anvraj	Bangalore	Trematodiasis	305 manoj		101
316	420 Shashi	Bangalore	Trematodiasis	308 manoj		101
317	421 Chandia	Bangalore	Dengue	306 manoj		101
318	422 Desmond	Pune	Trachoma	307 manoj		101
319	423 Kian	Venar	Dengue	311 manoj		101
320	424 Lami	Pune	Chikungunya	307 manoj		101
321	425 Adish	Chikmagalore	Malaria	307 manoj		101
322	426 Mohan	Belthangady	Rabies	309 manoj		101
323	456 Vishal	Mangalore	Leprony	302 manoj		101
324	491 John	Mangalore	Dengue	333 manoj		101

Create different views:

The screenshot shows a SQL Editor window with the following queries:

```

from patient as H
cross join Doctors as Doc

-----creating views---
create view new_healthcare
as select * from doctors h

select * from new_healthcare
  
```

The results pane displays a table with 12 rows and 4 columns:

Doctor_id	Doctor_Name	Department_id	Doctor_Mobile_Number
1	302 mayra	103	54.673.832
2	302 mahesh	104	54.673.832
3	303 swetha	105	54.673.832
4	304 bharvi	107	54.673.832
5	305 niti	111	54.673.832
6	306 pooja	121	54.671.832
7	307 khavya	151	54.673.832
8	308 sakshi	171	54.673.832
9	309 reba	131	54.673.832
10	211 kirti	111	54.673.832
11	332 anu	105	54.673.832
12	333 manoj	101	54.673.832

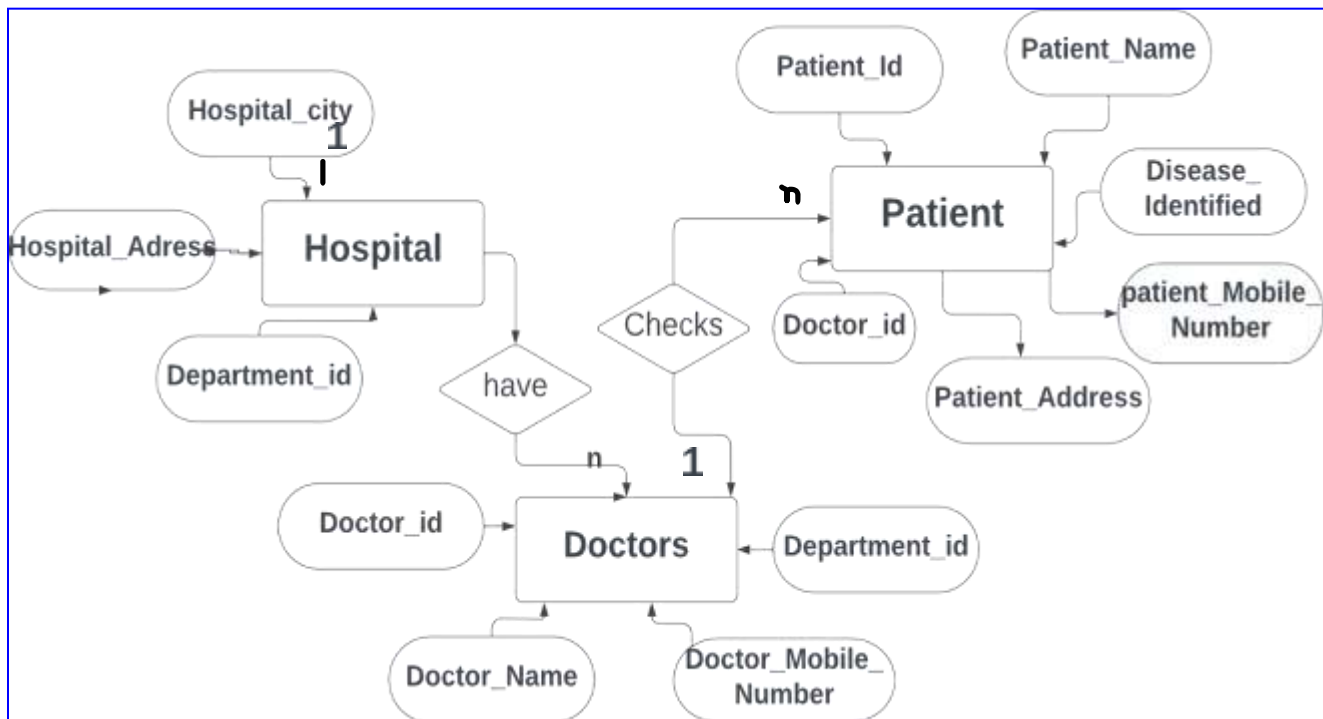
Creating view using join operation

```

create view healthcare
as
select H.Patient_Id ,H.Patient_Name ,H.Patient_Address,H.Disease_Identified,H.Doctor_id,
Doc.Doctor_name,Doc.Department_id
from patient as H
full join Doctors as Doc
on H.Doctor_id=Doc.Doctor_id

select * from healthcare
  
```

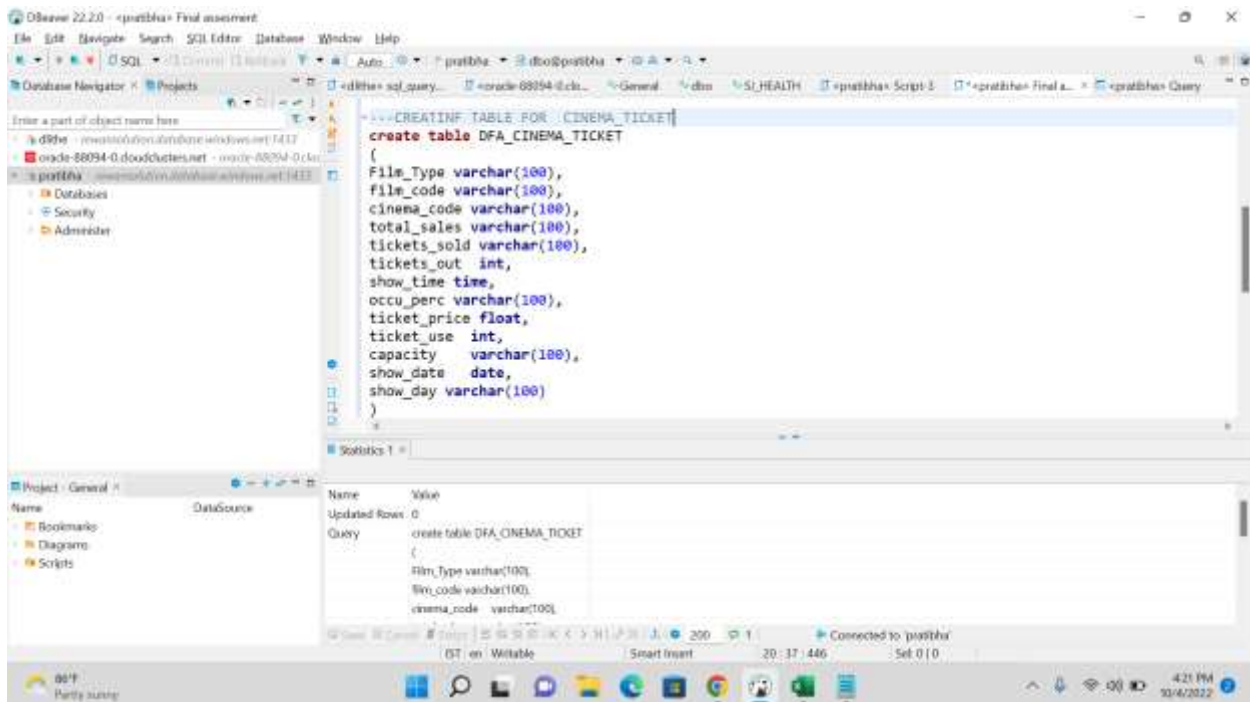
Patient_id	Patient_Name	Patient_Address	Disease_Identified	Doctor_id	Doctor_name	Department_id
401	Meppi	Bangalore	Legion	301	rajani	103
402	idile	Moodbadri	Chikungunya	305	nti	111
403	Gra	Mumbai	Malaria	302	maresh	104
404	Ja	Venur	Rabies	304	bavana	107
405	Mia	Chikmagalur	Snakebite	307	ibhaya	151
406	Shadu	Moodbadri	Tschoma	303	swathi	105
407	Ravi	Mangalore	Leprosy	307	ibhaya	151
408	Mykova	Ulle	Sabies	304	bavana	107
409	Mai	Venur	Tumutodis	305	nti	111
410	Marej	Balthargady	Dengue	301	rajani	103
411	Randy	Pure	Chikungunya	303	swathi	105
412	Jorge	Mangalore	Rabies	306	praja	121
413	Livanya	Ulle	Sabies	308	sakshi	171



2. Using transaction table perform various operations (DDL,DML)

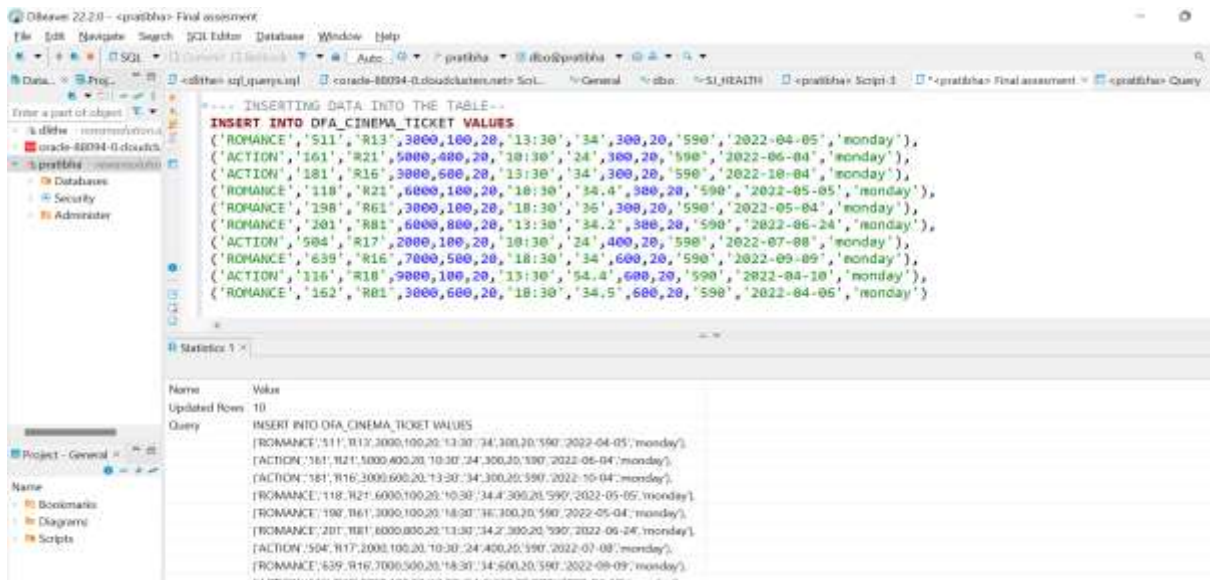
Answer:

Created table for cinema ticket.

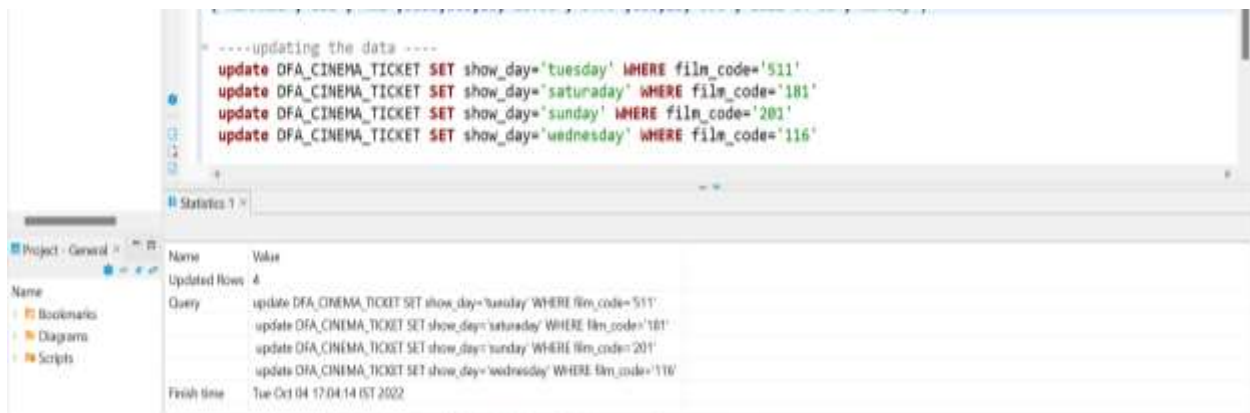


DML COMMANDS

INSERTING DATA INTO THE TABLE



Below snapshot shows the updating the table using where condition



Deleting the data using delete command



Displaying the data using select command

The screenshot shows the DBeaver SQL editor with the following SQL code:

```
update DFA_CINEMA_TICKET SET show_day='wednesday' WHERE film_code='116'

-----displaying data using select keyword--
select * from DFA_CINEMA_TICKET
```

The results of the query are displayed in a table with the following columns: film_type, film_code, cinema_code, total_sales, tickets_sold, tickets_out, show_time, occu_perc, ticket_price, ticket_use, and capacity. The table contains 11 rows of data.

film_type	film_code	cinema_code	total_sales	tickets_sold	tickets_out	show_time	occu_perc	ticket_price	ticket_use	capacity
ROMANCE	111	R11	3,000	100	20	13:30:00	34	300	20	590
ACTION	511	R13	3,000	100	20	13:30:00	34	300	20	590
ACTION	161	R21	5,000	400	20	10:30:00	24	300	20	590
ACTION	181	R16	3,000	600	20	13:30:00	34	300	20	590
ROMANCE	118	R21	6,000	100	20	10:30:00	34.4	300	20	590
ROMANCE	158	R61	3,000	100	20	18:30:00	36	300	20	590
ROMANCE	201	R81	6,000	200	20	13:30:00	34.2	300	20	590
ACTION	504	R17	2,000	100	20	10:30:00	24	400	20	590
ROMANCE	639	R16	7,000	500	20	18:30:00	34	600	20	590
ACTION	116	R18	8,000	100	20	13:30:00	54.4	600	20	590
ROMANCE	162	R01	3,000	600	20	18:30:00	34.5	600	20	590

DML COMMANDS

Adding new row using ALTER COMMAND

The screenshot shows the DBeaver SQL editor with the following SQL code:

```
-----DML COMMANDS-----
--adding new row using ALTER command
ALTER TABLE DFA_CINEMA_TICKET ADD cinema_name varchar(100);
```

The results of the command are displayed in a table with the following columns: Name, Value, and Statistics. The table contains 1 row of data.

Name	Value	Statistics
Updated Rows	0	

The query execution statistics are also displayed:

Query	Statistics
-----DML COMMANDS-----	
--adding new row using ALTER command	
ALTER TABLE DFA_CINEMA_TICKET ADD cinema_name varchar(100);	
Finish time	Thu Oct 06 11:11:05 IST 2022

Result of alter command, below snapshot shows the cinema_name column added to the table

delete from DFA_CINEMA_TICKET where film_code='116'

----DDL COMMENTS----

---adding new row using ALTER command

ALTER TABLE DFA_CINEMA_TICKET ADD cinema_name varchar(100);

select * from DFA_CINEMA_TICKET

	total_sales	tickets_sold	tickets_out	show_time	seats_per_show	ticket_price	ticket_use	capacity	show_date	show_day	cinema_name
1	3,000	100	20	13:30:00	34	300	20	590	2022-04-04	monday	P1011
2	3,000	100	20	13:30:00	34	300	20	590	2022-04-05	tuesday	P1011
3	5,000	400	20	10:30:00	24	300	20	590	2022-05-04	monday	P1011
4	3,000	600	20	13:30:00	34	300	20	590	2022-10-04	saturday	P1011
5	6,000	100	20	10:30:00	34.4	300	20	590	2022-05-05	monday	P1011
6	3,000	100	20	18:30:00	36	300	20	590	2022-05-04	monday	P1011
7	6,000	800	20	13:30:00	34.2	300	20	590	2022-06-24	sunday	P1011
8	2,000	100	20	10:30:00	24	400	20	590	2022-07-08	monday	P1011
9	7,000	500	20	18:30:00	34	400	20	590	2022-09-09	monday	P1011
10	1,000	600	20	18:30:00	34.5	600	20	590	2022-04-06	monday	P1011

Truncate table

delete from DFA_CINEMA_TICKET where film_code='116'

----DDL COMMENTS----

---adding new row using ALTER command

ALTER TABLE DFA_CINEMA_TICKET ADD cinema_name varchar(100);

select * from DFA_CINEMA_TICKET

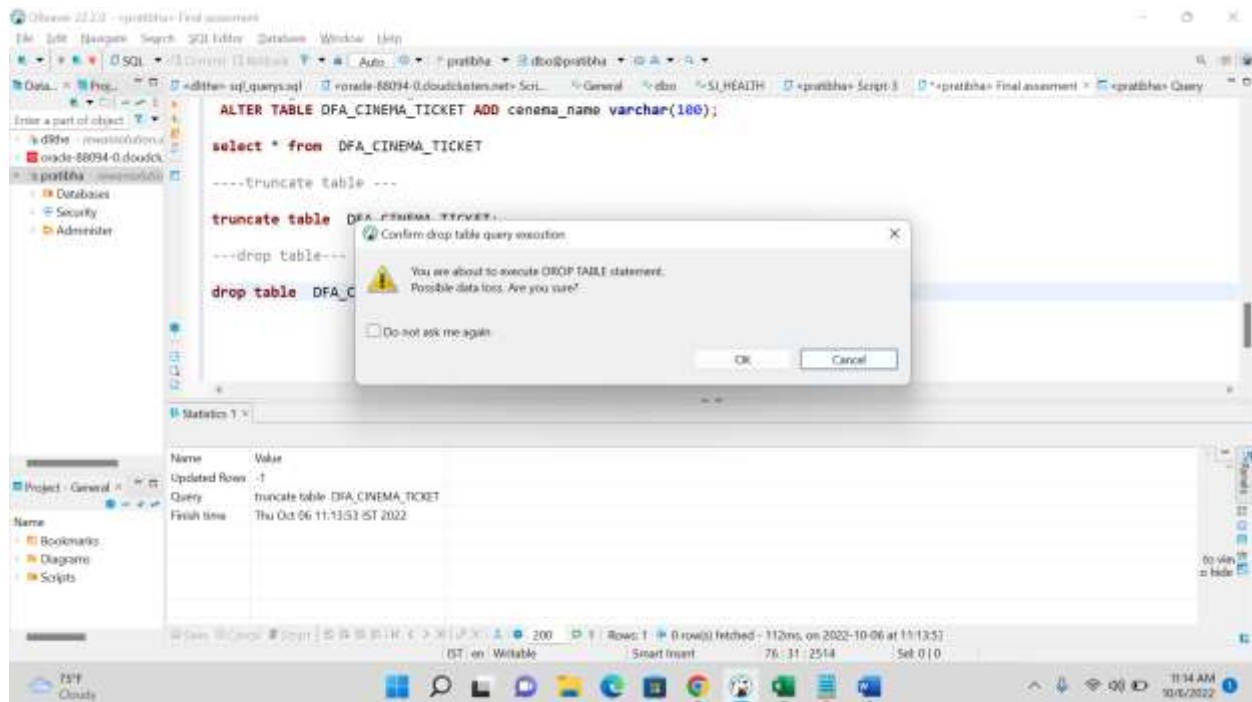
---truncate table---

truncate table DFA_CINEMA_TICKET;

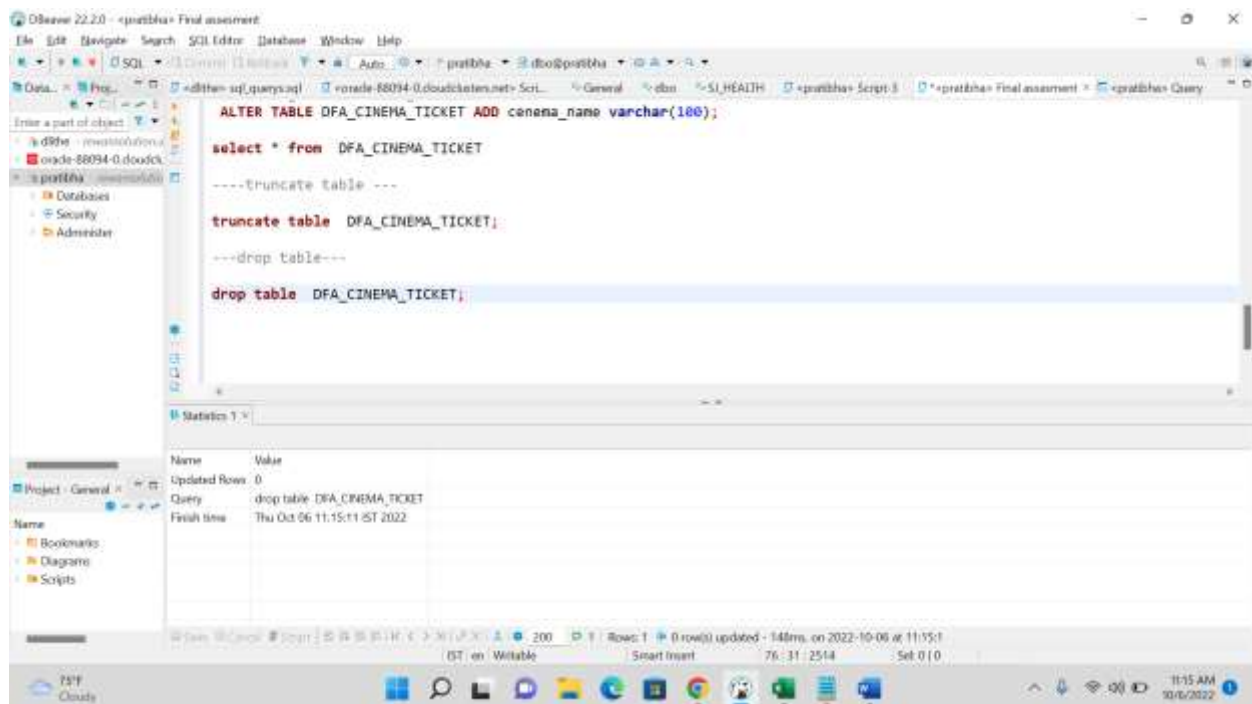
Name	Value
Updated Rows	-1
Query	truncate table DFA_CINEMA_TICKET
Finish time	Thu Oct 06 11:13:53 IST 2022

Drop table

While performing the drop table operation we will get a warning window



Below snapshot show the result of drop table



2nd data source

Creating covid healthcare table to perform ddl dml operations

```

--creating table for covid health --
CREATE TABLE covid_heath_care (
  DRGID varchar(50),
  DRG_Definition varchar(255),
  Provider_Id varchar(50),
  Provider_Name varchar(255),
  ProviderStreet_Address varchar(255),
  Provider_City varchar(255),
  Provider_State varchar(255),
  Provider_Zip_Code varchar(50),
  Hospital_Referral_Region_Description varchar(255),
  Total_Discharges varchar(50),
  Average_Covered_Charges varchar(50),
  Average_TotalPayments varchar(50),
  AverageMedicarePayments varchar(50)
)

-- view columns in table --
select * from covid_heath_care

-- DML COMMANDS --

-- updating the data --
update covid_heath_care SET Provider_Id='180' WHERE DRGID='45'
  
```

DDL commands

```

-- DML COMMANDS --

-- updating the data --
update covid_heath_care SET Provider_Id='180' WHERE DRGID='45'
update covid_heath_care SET Provider_Id='10001' WHERE DRGID='45'
update covid_heath_care SET Provider_Id='18003' WHERE DRGID='87'

-- displaying data using select keyword --
select * from covid_heath_care


-- DELETING THE DATA --
delete from covid_heath_care where provider_id='100'

-- DDL COMMANDS --
-- adding new row using ALTER command
ALTER TABLE covid_heath_care ADD disease_name varchar(100);

select * from covid_heath_care

-- truncate table --
truncate table covid_heath_care;
  
```


Below snapshot show the DDL commands



The screenshot shows the DBeaver SQL editor interface. The left sidebar displays a tree view of the database schema, including tables like 'Insuren', 'PS_BAP', 'PS_DEP', 'PS_ENE', 'PS_SCH', 'PS_SYN', 'PS_THR', 'PS_USE', 'PS_URG', 'agdrop', 'cinema', 'covid_h', and 'den_en'. The main editor window contains the following SQL commands:

```
select * from covid_heath_care

---DELETING THE DATA ---

delete from covid_heath_care where provider_id='100'

---DDL COMMANDS---
---adding new row using ALTER command
ALTER TABLE covid_heath_care ADD disease_name varchar(100);

select * from covid_heath_care

---truncate table ---

truncate table covid_heath_care;

---drop table---

drop table covid_heath_care;
```