MULTIMEDIA UNIVERSITY ®

TIS1101 Database Fundamentals

Assignment 2

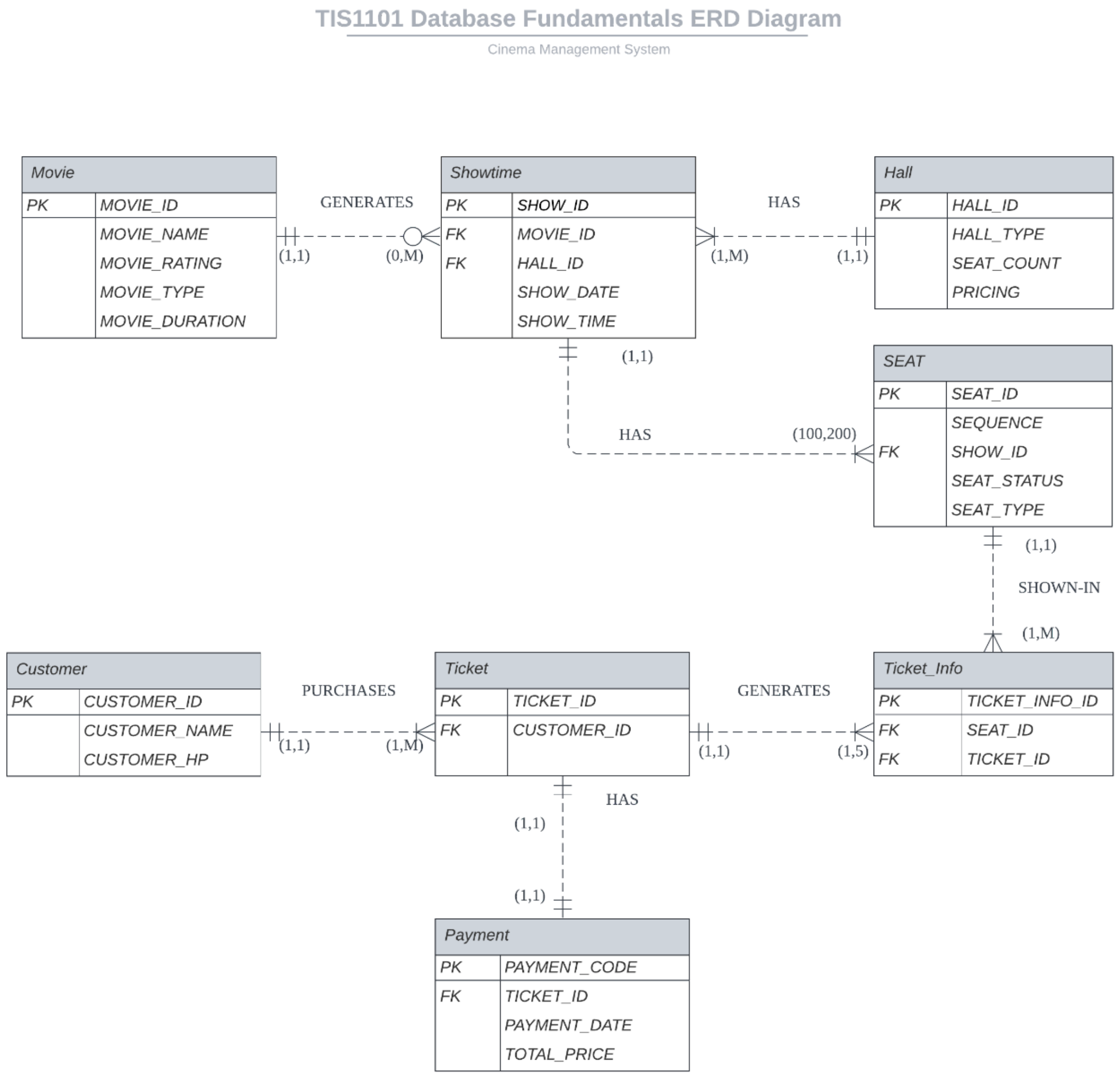
Title: Cinema Management System

Prepared by:

Leader:	1211102687	Emily Phang Ru Ying	1211102687@student.mmu.edu.my
Member:	1211102751	Teo Yu Jie	1211102751@student.mmu.edu.my
	1211102753	Lim Cai Qing	1211102753@student.mmu.edu.my

TABLE OF CONTENT

A. Corrected and normalized ERD



B.Data Dictionary

TABLE NAME	ATTRIBUTE NAME	CONTENTS	TYPE	REQUIRED	PK OR FK	FK REFERENCED TABLE
MOVIE	MOVIE_ID	Movie's ID	varchar(5)	Y	PK	
	MOVIE_NAME	Movie's name	varchar(50)	Y		
	MOVIE_RATING	Movie's rating	decimal(3,1)	Y		
	MOVIE_TYPE	Movie's genre	varchar(50)	Y		
	MOVIE_DURATION	Movie's duration in minutes	int	Y		
HALL	HALL_ID	Hall's ID	char(1)	Y	PK	
	HALL_TYPE	Hall's Type	varchar(10)	Y		
	SEAT_COUNT	Number of seats in the hall	int			
	PRICING	Hall's price based on hall type	decimal(5,2)	Y		
SHOWTIME	SHOW_ID	Show's ID	varchar(5)	Y	PK	

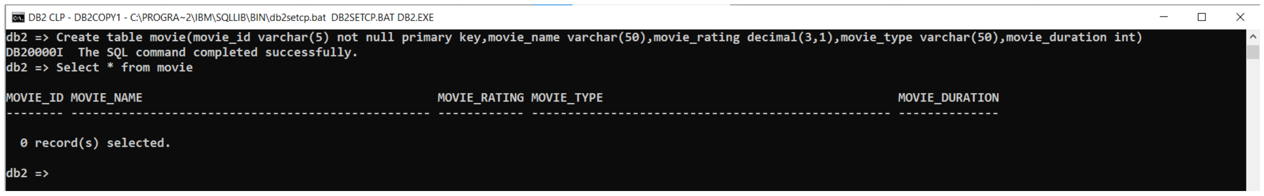
	MOVIE_ID	Movie's ID	varchar(5)	Y	FK	MOVIE
	HALL_ID	Hall's ID	char(1)	Y	FK	HALL
	SHOW_DATE	Date of the Show	date	Y		
	SHOW_TIME	Start time of the Show	time	Y		
SEAT	SEAT_ID	Seat's ID	int	Y	PK	
	SEQUENCE	Seat number in one showtime	int	Y		
	SHOW_ID	Show's ID	varchar(5)	Y	FK	SHOWTIME
	SEAT_STATUS	Booking status of the seat	varchar(10)	Y		
	SEAT_TYPE	Type of Seat	varchar(10)	Y		
CUSTOMER	CUSTOMER_ID	Customer's ID	varchar(10)	Y	PK	
	CUSTOMER_NAME	Customer's name	varchar(20)	Y		
	CUSTOMER_HP	Customer’s	bigint	Y		

		handphone number				
TICKET	TICKET_ID	Ticket's ID	varchar(5)	Y	PK	
	CUSTOMER_ID	Customer's ID	varchar(10)	Y	FK	CUSTOMER
TICKET_INFO	TICKET_INFO_ID	Ticket's Information ID	varchar(5)	Y	PK	
	SEAT_ID	Seat's ID	int	Y	FK	SEAT
	TICKET_ID	Ticket's ID	varchar(5)	Y	FK	TICKET
PAYMENT	PAYMENT_CODE	Payment Code	int	Y	PK	
	TICKET_ID	Ticket's ID	varchar(5)	Y	FK	TICKET
	PAYMENT_DATE	Date of Payment	date	Y		
	TOTAL_PRICE	TOTAL PRICE	decimal(7,2)	Y		

C.Creation Of Tables

1)CREATE TABLE MOVIE

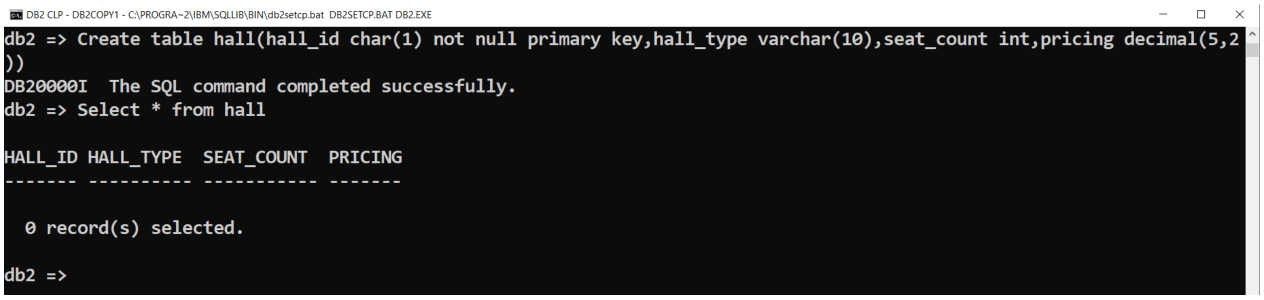
```
Create table movie
(
movie_id varchar(5) not null primary key,
movie_name varchar(50),
movie_rating decimal(3,1),
movie_type varchar(50),
movie_duration int
)
```



```
db2 => Create table movie(movie_id varchar(5) not null primary key,movie_name varchar(50),movie_rating decimal(3,1),movie_type varchar(50),movie_duration int)
DB20000I  The SQL command completed successfully.
db2 => Select * from movie

MOVIE_ID MOVIE_NAME                                         MOVIE_RATING MOVIE_TYPE                                         MOVIE_DURATION
-------- -------------------------------------------------- ------------ -------------------------------------------------- --------------

  0 record(s) selected.

db2 =>
```

2) CREATE TABLE HALL

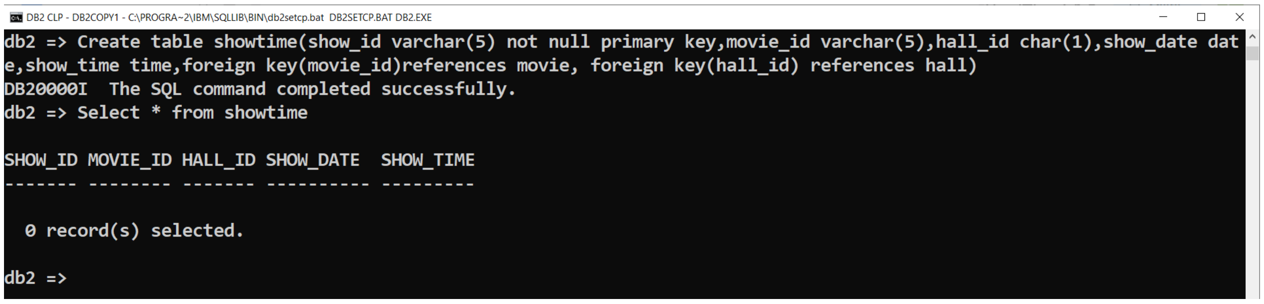
```
Create table hall
(
hall_id char(1) not null primary key,
hall_type varchar(10),
seat_count int,
pricing decimal(5,2)
)
```



```
db2 => Create table hall(hall_id char(1) not null primary key,hall_type varchar(10),seat_count int,pricing decimal(5,2
))
DB20000I  The SQL command completed successfully.
db2 => Select * from hall

HALL_ID HALL_TYPE  SEAT_COUNT  PRICING
------- ---------- ----------- -------

  0 record(s) selected.

db2 =>
```

3) CREATE TABLE SHOWTIME

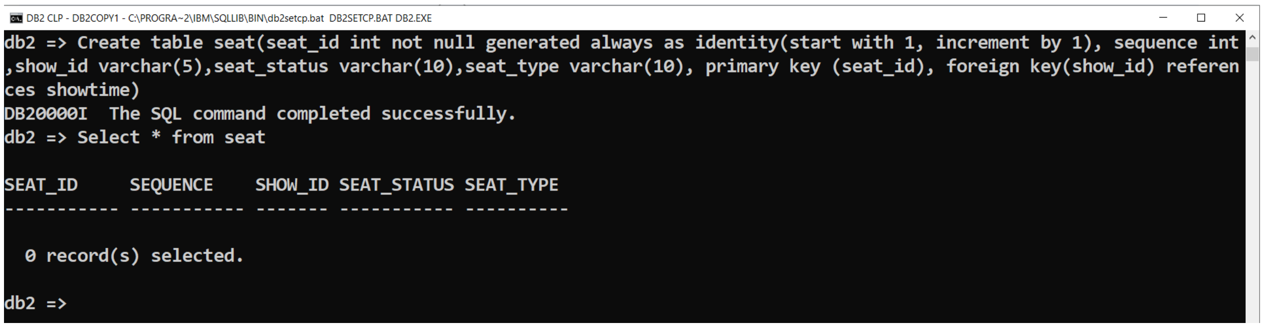
Create table showtime
(
show_id varchar(5) not null primary key,
movie_id varchar(5),
hall_id char(1),
show_date date,
show_time time,
foreign key(movie_id)references movie,
foreign key(hall_id) references hall
)


DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => Create table showtime(show_id varchar(5) not null primary key,movie_id varchar(5),hall_id char(1),show_date dat
e,show_time time,foreign key(movie_id)references movie, foreign key(hall_id) references hall)
DB20000I  The SQL command completed successfully.
db2 => Select * from showtime

SHOW_ID MOVIE_ID HALL_ID SHOW_DATE  SHOW_TIME
------- -------- ------- ---------- ---------

  0 record(s) selected.

db2 =>
```

4) CREATE TABLE SEAT

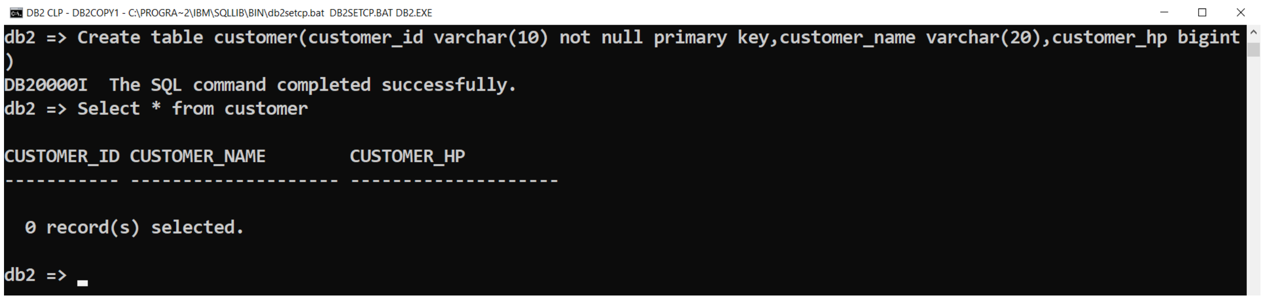
Create table seat
(
seat_id int not null generated always as identity(start with 1, increment by 1),
sequence int,
show_id varchar(5),
seat_status varchar(10),
seat_type varchar(10),
primary key (seat_id),
foreign key(show_id) references showtime
)


DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => Create table seat(seat_id int not null generated always as identity(start with 1, increment by 1), sequence int
,show_id varchar(5),seat_status varchar(10),seat_type varchar(10), primary key (seat_id), foreign key(show_id) referen
ces showtime)
DB20000I  The SQL command completed successfully.
db2 => Select * from seat

SEAT_ID     SEQUENCE    SHOW_ID SEAT_STATUS SEAT_TYPE
----------- ----------- ------- ----------- ----------

  0 record(s) selected.

db2 =>
```

5) CREATE TABLE CUSTOMER

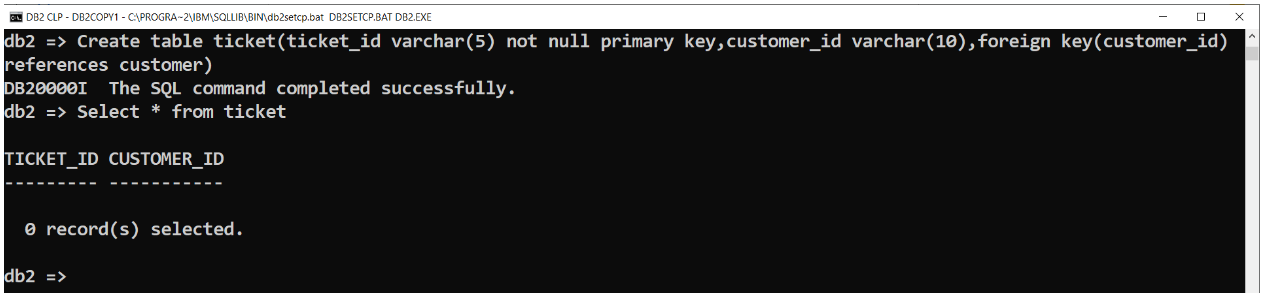
Create table customer
(
customer_id varchar(10) not null primary key,
customer_name varchar(20),
customer_hp bigint
)


DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => Create table customer(customer_id varchar(10) not null primary key,customer_name varchar(20),customer_hp bigint
)
DB20000I  The SQL command completed successfully.
db2 => Select * from customer

CUSTOMER_ID CUSTOMER_NAME        CUSTOMER_HP
----------- -------------------- --------------------

  0 record(s) selected.

db2 =>
```

6) CREATE TABLE TICKET

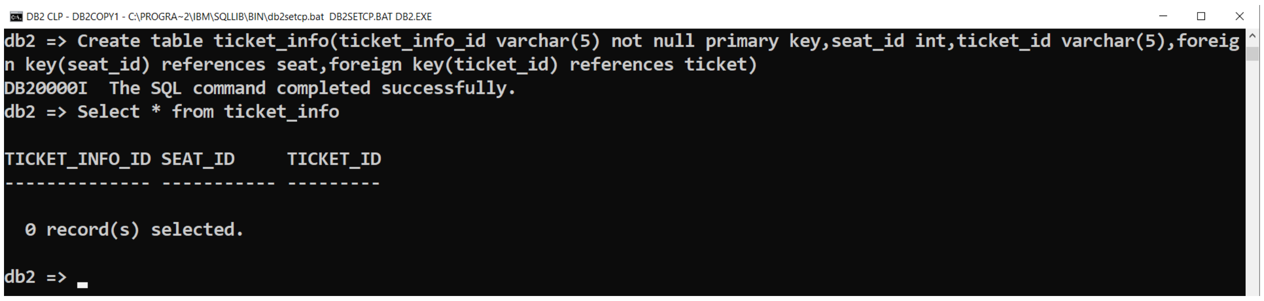
Create table ticket
(
ticket_id varchar(5) not null primary key,
customer_id varchar(10),
foreign key(customer_id) references customer
)


DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => Create table ticket(ticket_id varchar(5) not null primary key,customer_id varchar(10),foreign key(customer_id)
references customer)
DB20000I  The SQL command completed successfully.
db2 => Select * from ticket

TICKET_ID CUSTOMER_ID
--------- -----------

  0 record(s) selected.

db2 =>
```

7) CREATE TABLE TICKET_INFO

```
Create table ticket_info
(
ticket_info_id varchar(5) not null primary key,
seat_id int,
ticket_id varchar(5),
foreign key(seat_id) references seat,
foreign key(ticket_id) references ticket
)
```

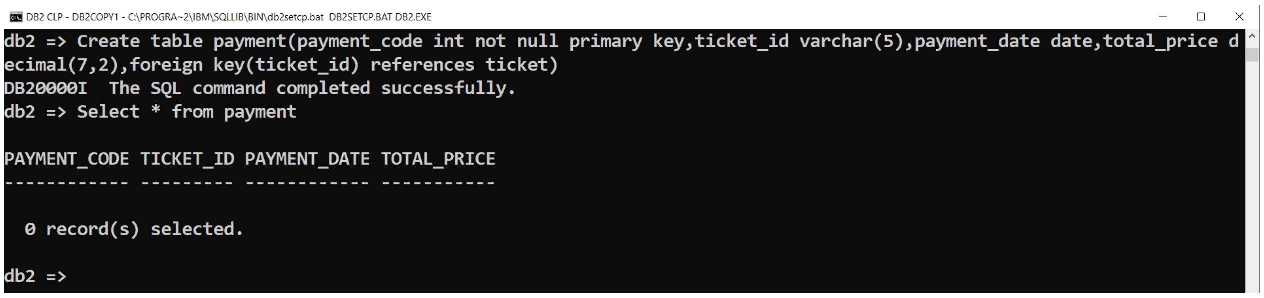


DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => Create table ticket_info(ticket_info_id varchar(5) not null primary key,seat_id int,ticket_id varchar(5),foreig
n key(seat_id) references seat,foreign key(ticket_id) references ticket)
DB20000I  The SQL command completed successfully.
db2 => Select * from ticket_info

TICKET_INFO_ID SEAT_ID     TICKET_ID
-------------- ----------- ---------

  0 record(s) selected.

db2 =>
```

8)CREATE TABLE PAYMENT

```
Create table payment
(
payment_code int not null primary key,
ticket_id varchar(5),
payment_date date,
total_price decimal(7,2),
foreign key(ticket_id) references ticket
)
```

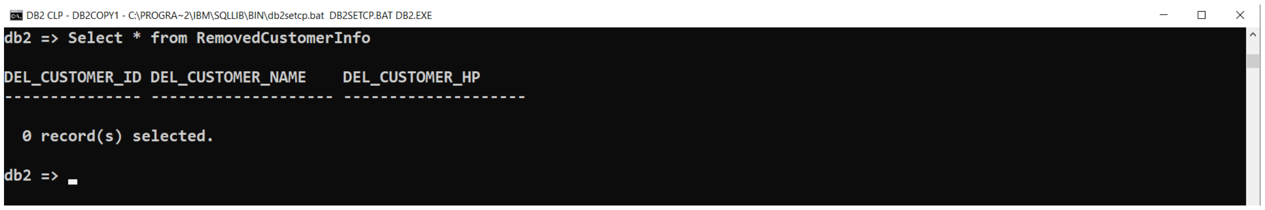


DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => Create table payment(payment_code int not null primary key,ticket_id varchar(5),payment_date date,total_price d
ecimal(7,2),foreign key(ticket_id) references ticket)
DB20000I  The SQL command completed successfully.
db2 => Select * from payment

PAYMENT_CODE TICKET_ID PAYMENT_DATE TOTAL_PRICE
------------ --------- ------------ -----------

  0 record(s) selected.

db2 =>
```

9)CREATE TABLE REMOVED CUSTOMER INFO

```
Create table RemovedCustomerInfo
(
Del_customer_id varchar(10),
Del_customer_name varchar(20),
Del_customer_hp bigint
)
```

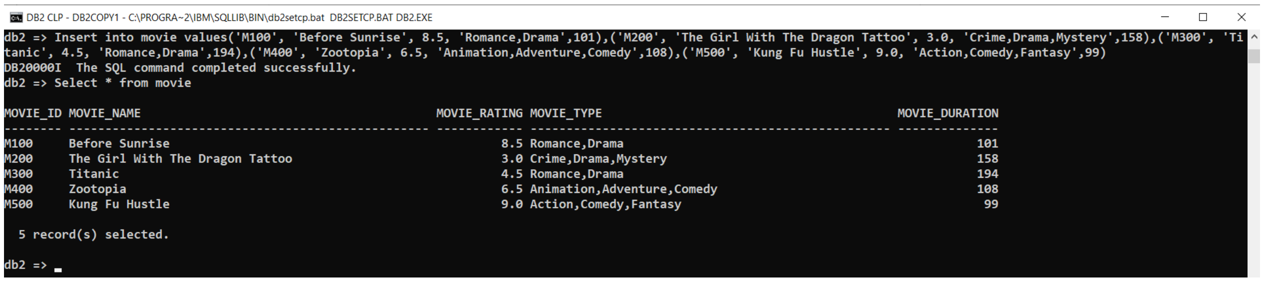


```
DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat  DB2SETCP.BAT DB2.EXE
db2 => Select * from RemovedCustomerInfo

DEL_CUSTOMER_ID DEL_CUSTOMER_NAME    DEL_CUSTOMER_HP
--------------- -------------------- --------------------

  0 record(s) selected.

db2 =>
```

D.Data Insertion

1) INSERT DATA INTO MOVIE TABLE

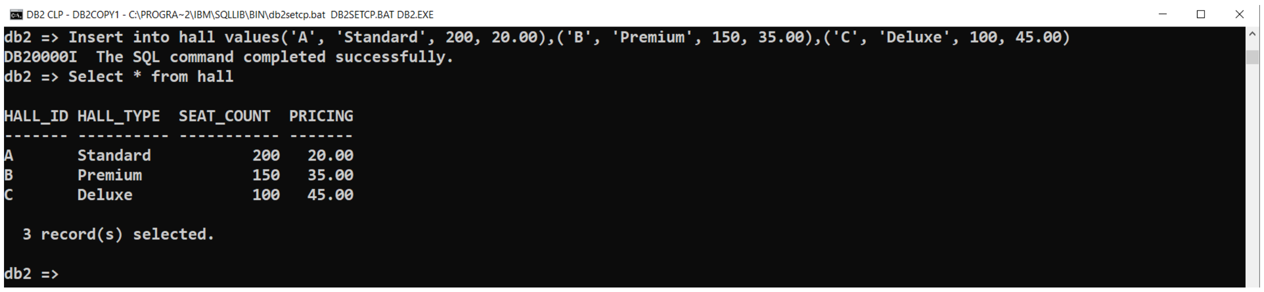
```
Insert into movie values
('M100', 'Before Sunrise', 8.5, 'Romance,Drama',101),
('M200', 'The Girl With The Dragon Tattoo', 3.0, 'Crime,Drama,Mystery',158),
('M300', 'Titanic', 4.5, 'Romance,Drama',194),
('M400', 'Zootopia', 6.5, 'Animation,Adventure,Comedy',108),
('M500', 'Kung Fu Hustle', 9.0, 'Action,Comedy,Fantasy',99)
```



```
DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat  DB2SETCP.BAT DB2.EXE
db2 => Insert into movie values('M100', 'Before Sunrise', 8.5, 'Romance,Drama',101),('M200', 'The Girl With The Dragon Tattoo', 3.0, 'Crime,Drama,Mystery',158),('M300', 'Ti
tanic', 4.5, 'Romance,Drama',194),('M400', 'Zootopia', 6.5, 'Animation,Adventure,Comedy',108),('M500', 'Kung Fu Hustle', 9.0, 'Action,Comedy,Fantasy',99)
DB20000I  The SQL command completed successfully.
db2 => Select * from movie

MOVIE_ID MOVIE_NAME                                         MOVIE_RATING MOVIE_TYPE                                         MOVIE_DURATION
-------- -------------------------------------------------- ------------ -------------------------------------------------- --------------
M100     Before Sunrise                                              8.5 Romance,Drama                                                 101
M200     The Girl With The Dragon Tattoo                             3.0 Crime,Drama,Mystery                                           158
M300     Titanic                                                     4.5 Romance,Drama                                                 194
M400     Zootopia                                                    6.5 Animation,Adventure,Comedy                                    108
M500     Kung Fu Hustle                                              9.0 Action,Comedy,Fantasy                                          99

  5 record(s) selected.

db2 => _
```

2) INSERT DATA INTO HALL TABLE

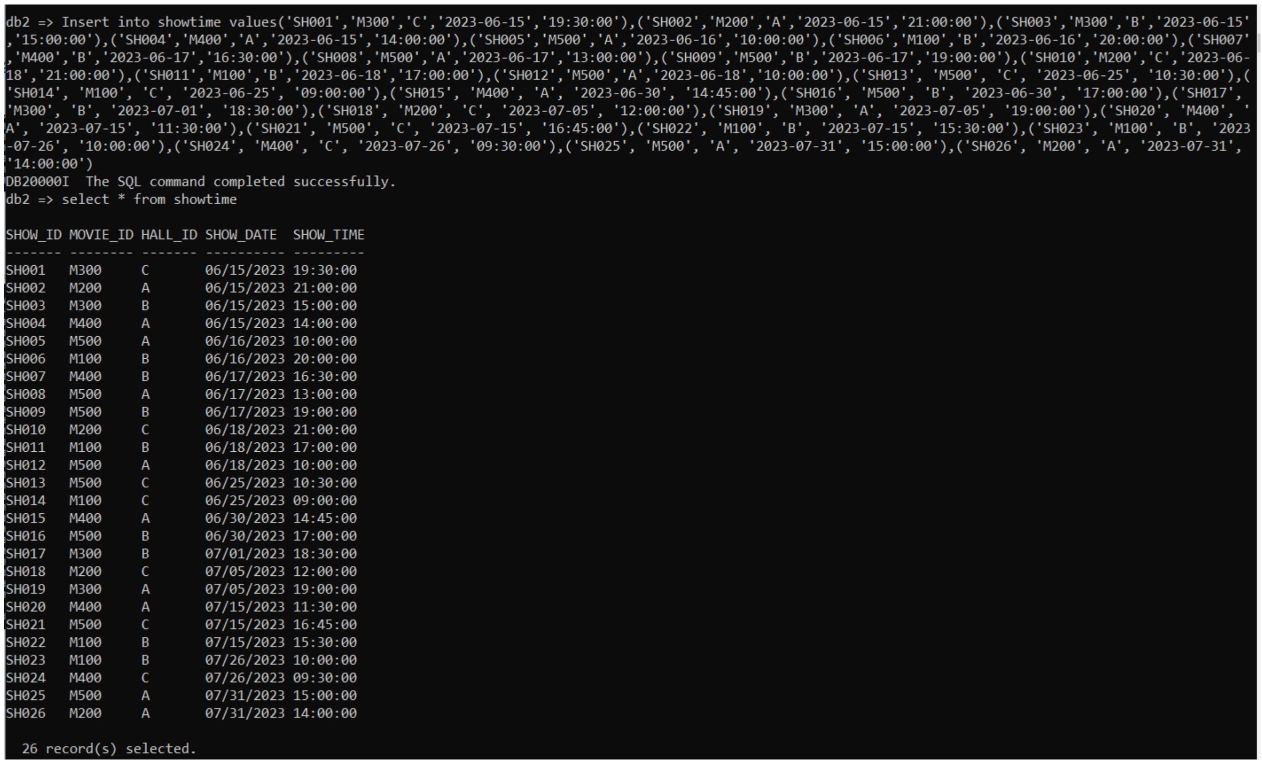
```
Insert into hall values
('A', 'Standard', 200, 20.00),
('B', 'Premium', 150, 35.00),
('C', 'Deluxe', 100, 45.00)
```



```
DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat  DB2SETCP.BAT DB2.EXE
db2 => Insert into hall values('A', 'Standard', 200, 20.00),('B', 'Premium', 150, 35.00),('C', 'Deluxe', 100, 45.00)
DB20000I  The SQL command completed successfully.
db2 => Select * from hall

HALL_ID HALL_TYPE  SEAT_COUNT  PRICING
------- ---------- ----------- -------
A       Standard           200   20.00
B       Premium            150   35.00
C       Deluxe             100   45.00

  3 record(s) selected.

db2 =>
```

3) INSERT DATA INTO SHOWTIME TABLE

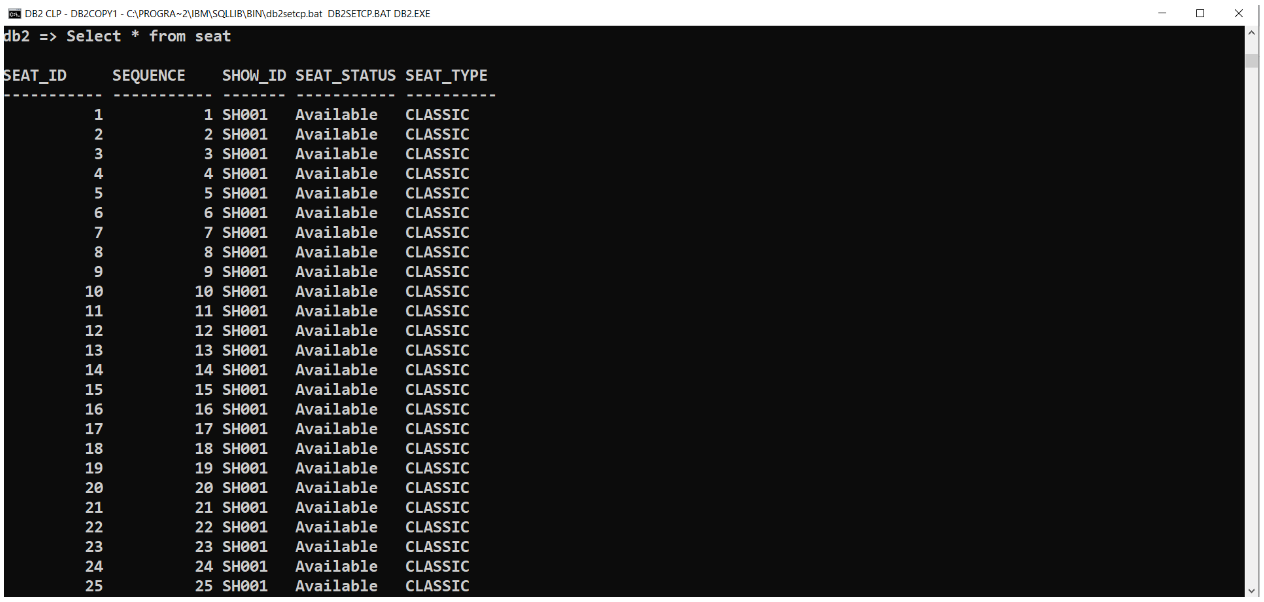
```
Insert into showtime values
('SH001','M300','C','2023-06-15','19:30:00'),
('SH002','M200','A','2023-06-15','21:00:00'),
('SH003','M300','B','2023-06-15','15:00:00'),
('SH004','M400','A','2023-06-15','14:00:00'),
('SH005','M500','A','2023-06-16','10:00:00'),
('SH006','M100','B','2023-06-16','20:00:00'),
('SH007','M400','B','2023-06-17','16:30:00'),
('SH008','M500','A','2023-06-17','13:00:00'),
('SH009','M500','B','2023-06-17','19:00:00'),
('SH010','M200','C','2023-06-18','21:00:00'),
('SH011','M100','B','2023-06-18','17:00:00'),
('SH012','M500','A','2023-06-18','10:00:00'),
('SH013', 'M500', 'C', '2023-06-25', '10:30:00'),
('SH014', 'M100', 'C', '2023-06-25', '09:00:00'),
('SH015', 'M400', 'A', '2023-06-30', '14:45:00'),
('SH016', 'M500', 'B', '2023-06-30', '17:00:00'),
('SH017', 'M300', 'B', '2023-07-01', '18:30:00'),
('SH018', 'M200', 'C', '2023-07-05', '12:00:00'),
('SH019', 'M300', 'A', '2023-07-05', '19:00:00'),
('SH020', 'M400', 'A', '2023-07-15', '11:30:00'),
('SH021', 'M500', 'C', '2023-07-15', '16:45:00'),
('SH022', 'M100', 'B', '2023-07-15', '15:30:00'),
('SH023', 'M100', 'B', '2023-07-26', '10:00:00'),
('SH024', 'M400', 'C', '2023-07-26', '09:30:00'),
('SH025', 'M500', 'A', '2023-07-31', '15:00:00'),
('SH026', 'M200', 'A', '2023-07-31', '14:00:00')
```



```
db2 => Insert into showtime values('SH001','M300','C','2023-06-15','19:30:00'),('SH002','M200','A','2023-06-15','21:00:00'),('SH003','M300','B','2023-06-15'
,'15:00:00'),('SH004','M400','A','2023-06-15','14:00:00'),('SH005','M500','A','2023-06-16','10:00:00'),('SH006','M100','B','2023-06-16','20:00:00'),('SH007'
,'M400','B','2023-06-17','16:30:00'),('SH008','M500','A','2023-06-17','13:00:00'),('SH009','M500','B','2023-06-17','19:00:00'),('SH010','M200','C','2023-06-
18','21:00:00'),('SH011','M100','B','2023-06-18','17:00:00'),('SH012','M500','A','2023-06-18','10:00:00'),('SH013', 'M500', 'C', '2023-06-25', '10:30:00'),(
'SH014', 'M100', 'C', '2023-06-25', '09:00:00'),('SH015', 'M400', 'A', '2023-06-30', '14:45:00'),('SH016', 'M500', 'B', '2023-06-30', '17:00:00'),('SH017',
'M300', 'B', '2023-07-01', '18:30:00'),('SH018', 'M200', 'C', '2023-07-05', '12:00:00'),('SH019', 'M300', 'A', '2023-07-05', '19:00:00'),('SH020', 'M400', '
A', '2023-07-15', '11:30:00'),('SH021', 'M500', 'C', '2023-07-15', '16:45:00'),('SH022', 'M100', 'B', '2023-07-15', '15:30:00'),('SH023', 'M100', 'B', '2023
-07-26', '10:00:00'),('SH024', 'M400', 'C', '2023-07-26', '09:30:00'),('SH025', 'M500', 'A', '2023-07-31', '15:00:00'),('SH026', 'M200', 'A', '2023-07-31',
'14:00:00')
DB20000I  The SQL command completed successfully.
db2 => select * from showtime

SHOW_ID MOVIE_ID HALL_ID SHOW_DATE  SHOW_TIME
------- -------- ------- ---------- ---------
SH001   M300     C       06/15/2023 19:30:00
SH002   M200     A       06/15/2023 21:00:00
SH003   M300     B       06/15/2023 15:00:00
SH004   M400     A       06/15/2023 14:00:00
SH005   M500     A       06/16/2023 10:00:00
SH006   M100     B       06/16/2023 20:00:00
SH007   M400     B       06/17/2023 16:30:00
SH008   M500     A       06/17/2023 13:00:00
SH009   M500     B       06/17/2023 19:00:00
SH010   M200     C       06/18/2023 21:00:00
SH011   M100     B       06/18/2023 17:00:00
SH012   M500     A       06/18/2023 10:00:00
SH013   M500     C       06/25/2023 10:30:00
SH014   M100     C       06/25/2023 09:00:00
SH015   M400     A       06/30/2023 14:45:00
SH016   M500     B       06/30/2023 17:00:00
SH017   M300     B       07/01/2023 18:30:00
SH018   M200     C       07/05/2023 12:00:00
SH019   M300     A       07/05/2023 19:00:00
SH020   M400     A       07/15/2023 11:30:00
SH021   M500     C       07/15/2023 16:45:00
SH022   M100     B       07/15/2023 15:30:00
SH023   M100     B       07/26/2023 10:00:00
SH024   M400     C       07/26/2023 09:30:00
SH025   M500     A       07/31/2023 15:00:00
SH026   M200     A       07/31/2023 14:00:00

  26 record(s) selected.
```

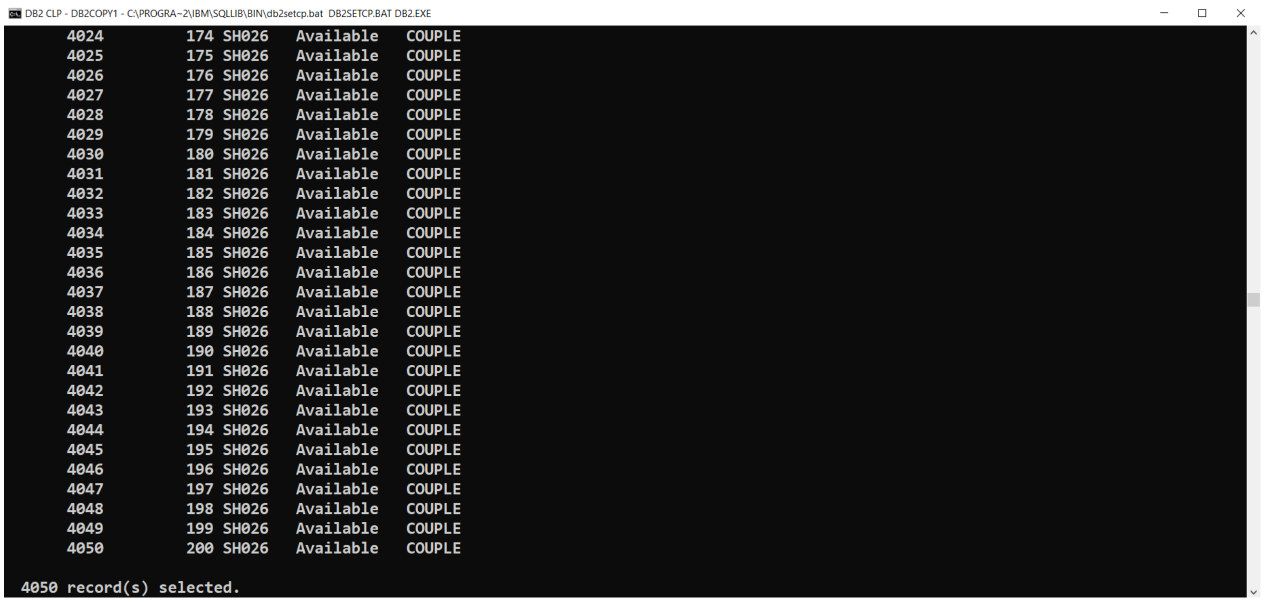
4) INSERT DATA INTO SEAT TABLE

The Seat_ID and sequence records in the seat table are automatically generated based on the showtime table. This process is facilitated by a combination of triggers and a procedure. The trg_showtime trigger is invoked when new records are inserted into the showtime table, and it in turn calls the auto_seat procedure. The auto_seat procedure retrieves the total seat count from the Hall table, and using a cursor, inserts the corresponding seat records into the seat table. Additionally, the trgSeatType trigger is triggered after inserts on the seat table, updating the seat_type column based on a calculation involving the seat count from the Hall and showtime tables. Furthermore, For more detailed information on these triggers and the auto_seat procedure, please refer to the respective trigger and procedure definitions.



DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => Select * from seat

SEAT_ID     SEQUENCE    SHOW_ID SEAT_STATUS SEAT_TYPE
----------- ----------- ------- ----------- ----------
          1           1 SH001   Available   CLASSIC
          2           2 SH001   Available   CLASSIC
          3           3 SH001   Available   CLASSIC
          4           4 SH001   Available   CLASSIC
          5           5 SH001   Available   CLASSIC
          6           6 SH001   Available   CLASSIC
          7           7 SH001   Available   CLASSIC
          8           8 SH001   Available   CLASSIC
          9           9 SH001   Available   CLASSIC
         10          10 SH001   Available   CLASSIC
         11          11 SH001   Available   CLASSIC
         12          12 SH001   Available   CLASSIC
         13          13 SH001   Available   CLASSIC
         14          14 SH001   Available   CLASSIC
         15          15 SH001   Available   CLASSIC
         16          16 SH001   Available   CLASSIC
         17          17 SH001   Available   CLASSIC
         18          18 SH001   Available   CLASSIC
         19          19 SH001   Available   CLASSIC
         20          20 SH001   Available   CLASSIC
         21          21 SH001   Available   CLASSIC
         22          22 SH001   Available   CLASSIC
         23          23 SH001   Available   CLASSIC
         24          24 SH001   Available   CLASSIC
         25          25 SH001   Available   CLASSIC
```

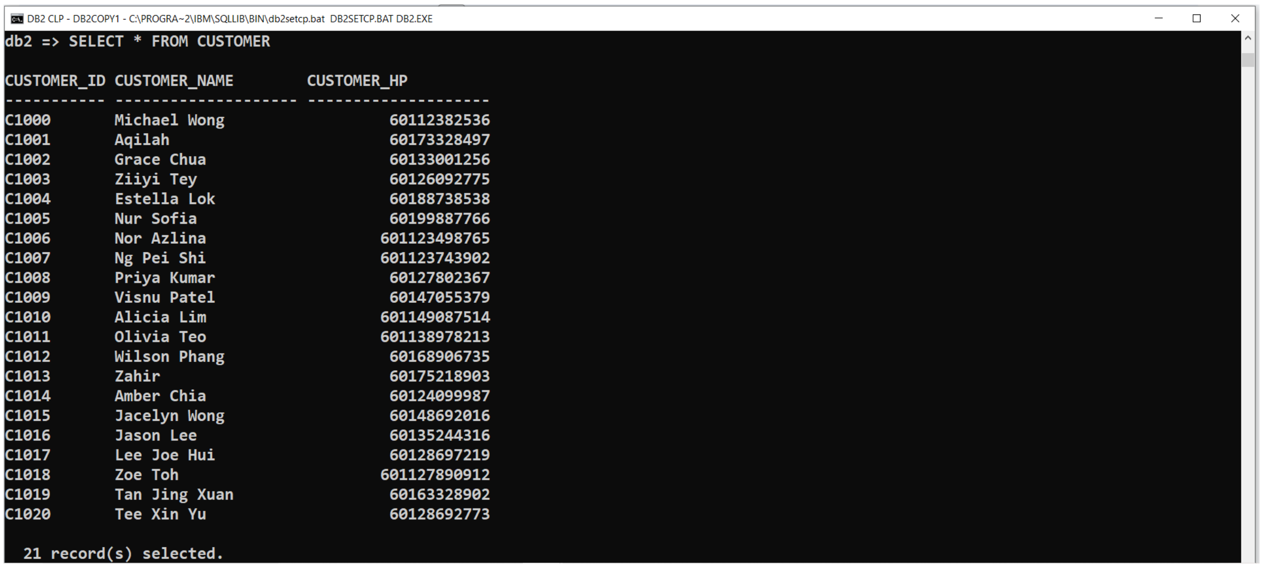


DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
       4024         174 SH026   Available   COUPLE
       4025         175 SH026   Available   COUPLE
       4026         176 SH026   Available   COUPLE
       4027         177 SH026   Available   COUPLE
       4028         178 SH026   Available   COUPLE
       4029         179 SH026   Available   COUPLE
       4030         180 SH026   Available   COUPLE
       4031         181 SH026   Available   COUPLE
       4032         182 SH026   Available   COUPLE
       4033         183 SH026   Available   COUPLE
       4034         184 SH026   Available   COUPLE
       4035         185 SH026   Available   COUPLE
       4036         186 SH026   Available   COUPLE
       4037         187 SH026   Available   COUPLE
       4038         188 SH026   Available   COUPLE
       4039         189 SH026   Available   COUPLE
       4040         190 SH026   Available   COUPLE
       4041         191 SH026   Available   COUPLE
       4042         192 SH026   Available   COUPLE
       4043         193 SH026   Available   COUPLE
       4044         194 SH026   Available   COUPLE
       4045         195 SH026   Available   COUPLE
       4046         196 SH026   Available   COUPLE
       4047         197 SH026   Available   COUPLE
       4048         198 SH026   Available   COUPLE
       4049         199 SH026   Available   COUPLE
       4050         200 SH026   Available   COUPLE

  4050 record(s) selected.
```

5) INSERT DATA INTO CUSTOMER TABLE

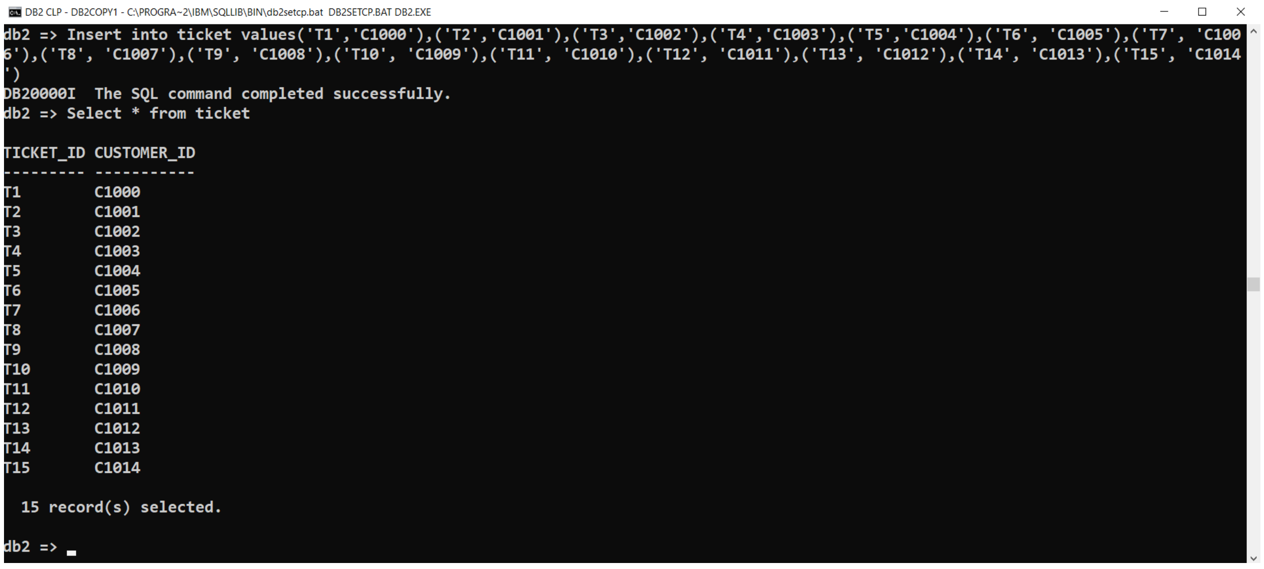
```
Insert into customer values
('C1000', 'Michael Wong', 60112382536),
('C1001', 'Aqilah', 60173328497),
('C1002', 'Grace Chua', 60133001256),
('C1003', 'Ziiyi Tey', 60126092775),
('C1004', 'Estella Lok', 60188738538),
('C1005', 'Nur Sofia',60199887766),
('C1006', 'Nor Azlina', 601123498765),
('C1007', 'Ng Pei Shi', 601123743902),
('C1008', 'Priya Kumar', 60127802367),
('C1009', 'Visnu Patel', 60147055379),
('C1010', 'Alicia Lim', 601149087514),
('C1011', 'Olivia Teo', 601138978213),
('C1012', 'Wilson Phang', 60168906735),
('C1013', 'Zahir', 60175218903),
('C1014','Amber Chia',60124099987),
('C1015', 'Jacelyn Wong', 60148692016),
('C1016','Jason Lee',60135244316),
('C1017', 'Lee Joe Hui', 60128697219),
('C1018','Zoe Toh',601127890912),
('C1019', 'Tan Jing Xuan', 60163328902),
('C1020','Tee Xin Yu',60128692773)
```


DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => SELECT * FROM CUSTOMER

CUSTOMER_ID CUSTOMER_NAME        CUSTOMER_HP
----------- -------------------- --------------------
C1000       Michael Wong                  60112382536
C1001       Aqilah                        60173328497
C1002       Grace Chua                    60133001256
C1003       Ziiyi Tey                     60126092775
C1004       Estella Lok                   60188738538
C1005       Nur Sofia                     60199887766
C1006       Nor Azlina                   601123498765
C1007       Ng Pei Shi                   601123743902
C1008       Priya Kumar                   60127802367
C1009       Visnu Patel                   60147055379
C1010       Alicia Lim                   601149087514
C1011       Olivia Teo                   601138978213
C1012       Wilson Phang                  60168906735
C1013       Zahir                         60175218903
C1014       Amber Chia                    60124099987
C1015       Jacelyn Wong                  60148692016
C1016       Jason Lee                     60135244316
C1017       Lee Joe Hui                   60128697219
C1018       Zoe Toh                      601127890912
C1019       Tan Jing Xuan                 60163328902
C1020       Tee Xin Yu                    60128692773

  21 record(s) selected.
```

6) INSERT DATA INTO TICKET TABLE

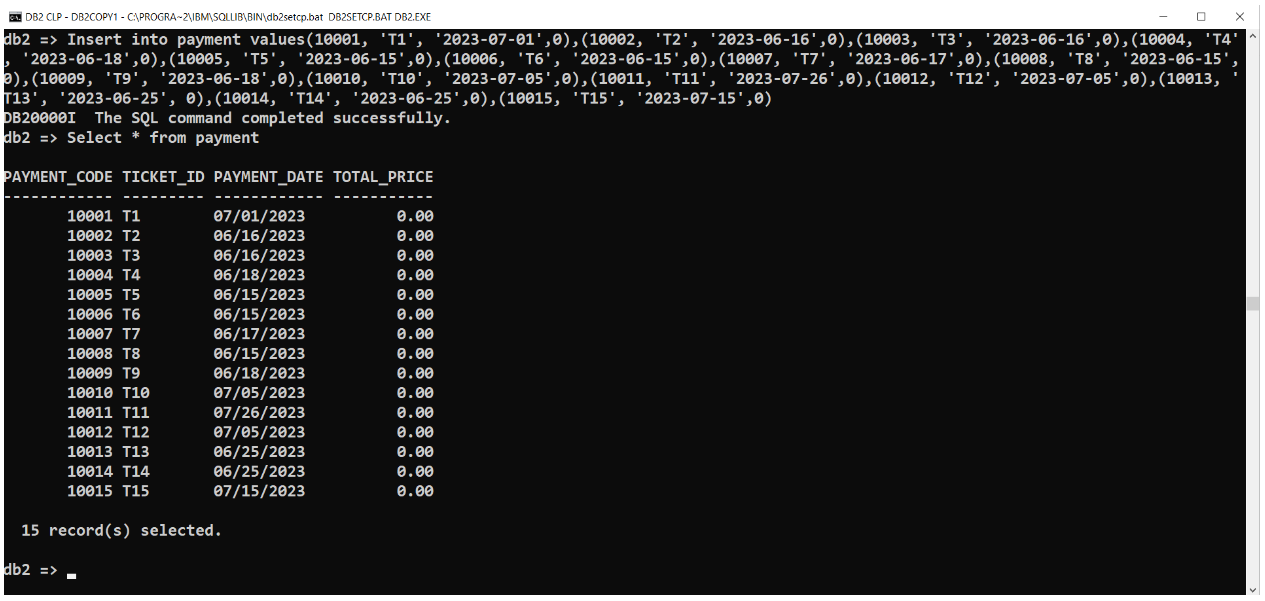
Insert into ticket values
('T1','C1000'),
('T2','C1001'),
('T3','C1002'),
('T4','C1003'),
('T5','C1004'),
('T6', 'C1005'),
('T7', 'C1006'),
('T8', 'C1007'),
('T9', 'C1008'),
('T10', 'C1009'),
('T11', 'C1010'),
('T12', 'C1011'),
('T13', 'C1012'),
('T14', 'C1013'),
('T15', 'C1014')


DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => Insert into ticket values('T1','C1000'),('T2','C1001'),('T3','C1002'),('T4','C1003'),('T5','C1004'),('T6', 'C1005'),('T7', 'C100
6'),('T8', 'C1007'),('T9', 'C1008'),('T10', 'C1009'),('T11', 'C1010'),('T12', 'C1011'),('T13', 'C1012'),('T14', 'C1013'),('T15', 'C1014
')
DB20000I  The SQL command completed successfully.
db2 => Select * from ticket

TICKET_ID CUSTOMER_ID
--------- -----------
T1        C1000
T2        C1001
T3        C1002
T4        C1003
T5        C1004
T6        C1005
T7        C1006
T8        C1007
T9        C1008
T10       C1009
T11       C1010
T12       C1011
T13       C1012
T14       C1013
T15       C1014

  15 record(s) selected.

db2 =>
```

7) INSERT DATA INTO PAYMENT TABLE

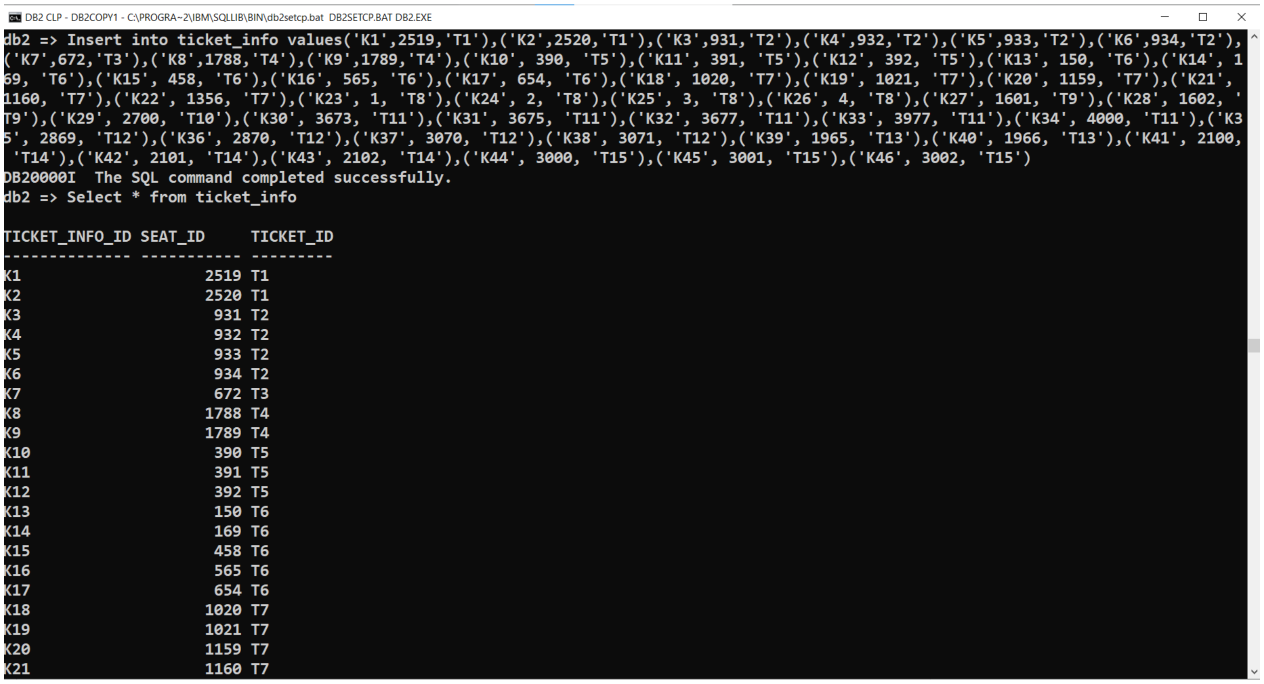
```
Insert into payment values
(10001, 'T1', '2023-07-01',0),
(10002, 'T2', '2023-06-16',0),
(10003, 'T3', '2023-06-16',0),
(10004, 'T4', '2023-06-18',0),
(10005, 'T5', '2023-06-15',0),
(10006, 'T6', '2023-06-15',0),
(10007, 'T7', '2023-06-17',0),
(10008, 'T8', '2023-06-15',0),
(10009, 'T9', '2023-06-18',0),
(10010, 'T10', '2023-07-05',0),
(10011, 'T11', '2023-07-26',0),
(10012, 'T12', '2023-07-05',0),
(10013, 'T13', '2023-06-25', 0),
(10014, 'T14', '2023-06-25',0),
(10015, 'T15', '2023-07-15',0)
```


DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => Insert into payment values(10001, 'T1', '2023-07-01',0),(10002, 'T2', '2023-06-16',0),(10003, 'T3', '2023-06-16',0),(10004, 'T4'
, '2023-06-18',0),(10005, 'T5', '2023-06-15',0),(10006, 'T6', '2023-06-15',0),(10007, 'T7', '2023-06-17',0),(10008, 'T8', '2023-06-15',
0),(10009, 'T9', '2023-06-18',0),(10010, 'T10', '2023-07-05',0),(10011, 'T11', '2023-07-26',0),(10012, 'T12', '2023-07-05',0),(10013, '
T13', '2023-06-25', 0),(10014, 'T14', '2023-06-25',0),(10015, 'T15', '2023-07-15',0)
DB20000I  The SQL command completed successfully.
db2 => Select * from payment

PAYMENT_CODE TICKET_ID PAYMENT_DATE TOTAL_PRICE
------------ --------- ------------ -----------
       10001 T1        07/01/2023          0.00
       10002 T2        06/16/2023          0.00
       10003 T3        06/16/2023          0.00
       10004 T4        06/18/2023          0.00
       10005 T5        06/15/2023          0.00
       10006 T6        06/15/2023          0.00
       10007 T7        06/17/2023          0.00
       10008 T8        06/15/2023          0.00
       10009 T9        06/18/2023          0.00
       10010 T10       07/05/2023          0.00
       10011 T11       07/26/2023          0.00
       10012 T12       07/05/2023          0.00
       10013 T13       06/25/2023          0.00
       10014 T14       06/25/2023          0.00
       10015 T15       07/15/2023          0.00

  15 record(s) selected.

db2 =>
```

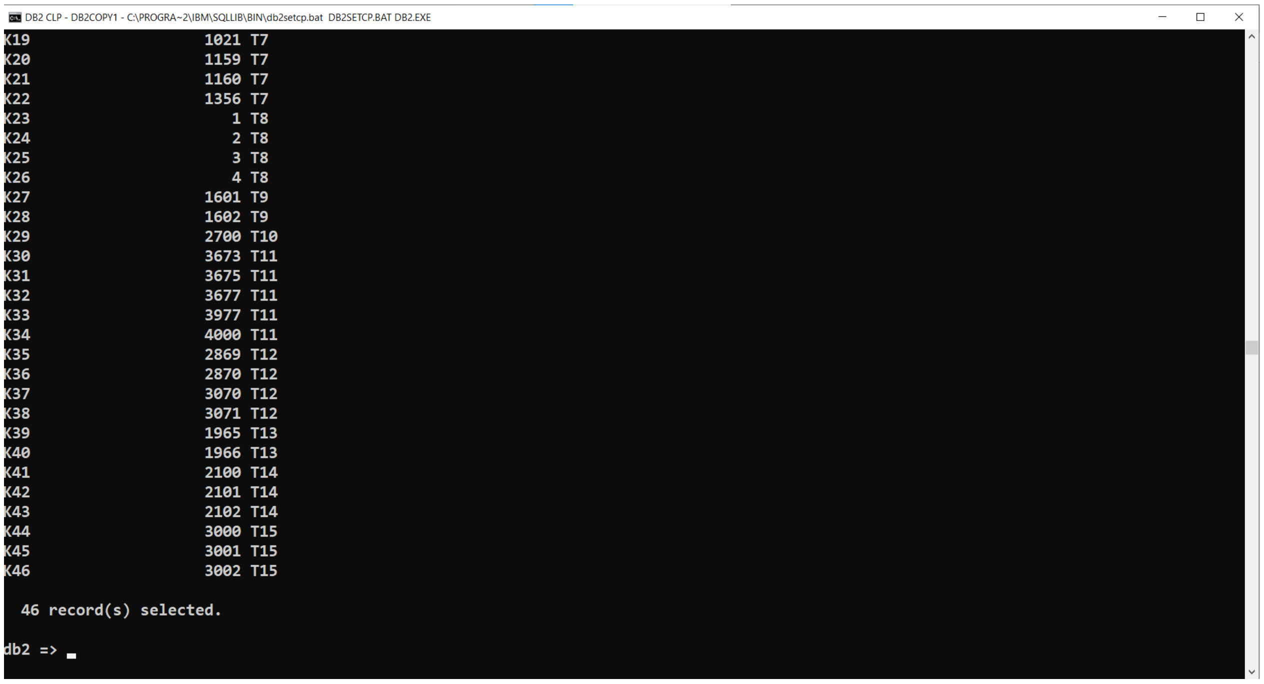
8) INSERT DATA INTO TICKET_INFO TABLE

```
Insert into ticket_info values
('K1',2519,'T1'),
('K2',2520,'T1'),
('K3',931,'T2'),
('K4',932,'T2'),
('K5',933,'T2'),
('K6',934,'T2'),
('K7',672,'T3'),
('K8',1788,'T4'),
('K9',1789,'T4'),
('K10', 390, 'T5'),
('K11', 391, 'T5'),
('K12', 392, 'T5'),
('K13', 150, 'T6'),
('K14', 169, 'T6'),
('K15', 458, 'T6'),
('K16', 565, 'T6'),
('K17', 654, 'T6'),
('K18', 1020, 'T7'),
('K19', 1021, 'T7'),
('K20', 1159, 'T7'),
('K21', 1160, 'T7'),
('K22', 1356, 'T7'),
('K23', 1, 'T8'),
('K24', 2, 'T8'),
('K25', 3, 'T8'),
('K26', 4, 'T8'),
('K27', 1601, 'T9'),
('K28', 1602, 'T9'),
('K29', 2700, 'T10'),
('K30', 3673, 'T11')
,('K31', 3675, 'T11'),
('K32', 3677, 'T11'),
('K33', 3977, 'T11'),
('K34', 4000, 'T11'),
('K35', 2869, 'T12'),
('K36', 2870, 'T12'),
('K37', 3070, 'T12'),
('K38', 3071, 'T12'),
('K39', 1965, 'T13'),
('K40', 1966, 'T13'),
('K41', 2100, 'T14'),
('K42', 2101, 'T14'),
('K43', 2102, 'T14'),
('K44', 3000, 'T15'),
('K45', 3001, 'T15'),
('K46', 3002, 'T15')
```



DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => Insert into ticket_info values('K1',2519,'T1'),('K2',2520,'T1'),('K3',931,'T2'),('K4',932,'T2'),('K5',933,'T2'),('K6',934,'T2'),
('K7',672,'T3'),('K8',1788,'T4'),('K9',1789,'T4'),('K10', 390, 'T5'),('K11', 391, 'T5'),('K12', 392, 'T5'),('K13', 150, 'T6'),('K14', 1
69, 'T6'),('K15', 458, 'T6'),('K16', 565, 'T6'),('K17', 654, 'T6'),('K18', 1020, 'T7'),('K19', 1021, 'T7'),('K20', 1159, 'T7'),('K21',
1160, 'T7'),('K22', 1356, 'T7'),('K23', 1, 'T8'),('K24', 2, 'T8'),('K25', 3, 'T8'),('K26', 4, 'T8'),('K27', 1601, 'T9'),('K28', 1602, '
T9'),('K29', 2700, 'T10'),('K30', 3673, 'T11'),('K31', 3675, 'T11'),('K32', 3677, 'T11'),('K33', 3977, 'T11'),('K34', 4000, 'T11'),('K3
5', 2869, 'T12'),('K36', 2870, 'T12'),('K37', 3070, 'T12'),('K38', 3071, 'T12'),('K39', 1965, 'T13'),('K40', 1966, 'T13'),('K41', 2100,
 'T14'),('K42', 2101, 'T14'),('K43', 2102, 'T14'),('K44', 3000, 'T15'),('K45', 3001, 'T15'),('K46', 3002, 'T15')
DB20000I  The SQL command completed successfully.
db2 => Select * from ticket_info

TICKET_INFO_ID SEAT_ID     TICKET_ID
-------------- ----------- ---------
K1                    2519 T1
K2                    2520 T1
K3                     931 T2
K4                     932 T2
K5                     933 T2
K6                     934 T2
K7                     672 T3
K8                    1788 T4
K9                    1789 T4
K10                    390 T5
K11                    391 T5
K12                    392 T5
K13                    150 T6
K14                    169 T6
K15                    458 T6
K16                    565 T6
K17                    654 T6
K18                   1020 T7
K19                   1021 T7
K20                   1159 T7
K21                   1160 T7
```



DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
K19                   1021 T7
K20                   1159 T7
K21                   1160 T7
K22                   1356 T7
K23                      1 T8
K24                      2 T8
K25                      3 T8
K26                      4 T8
K27                   1601 T9
K28                   1602 T9
K29                   2700 T10
K30                   3673 T11
K31                   3675 T11
K32                   3677 T11
K33                   3977 T11
K34                   4000 T11
K35                   2869 T12
K36                   2870 T12
K37                   3070 T12
K38                   3071 T12
K39                   1965 T13
K40                   1966 T13
K41                   2100 T14
K42                   2101 T14
K43                   2102 T14
K44                   3000 T15
K45                   3001 T15
K46                   3002 T15

  46 record(s) selected.

db2 =>
```

E.Data Manipulation With SQL

☐ Stored Procedure

Procedure Name:auto_seat

Description:

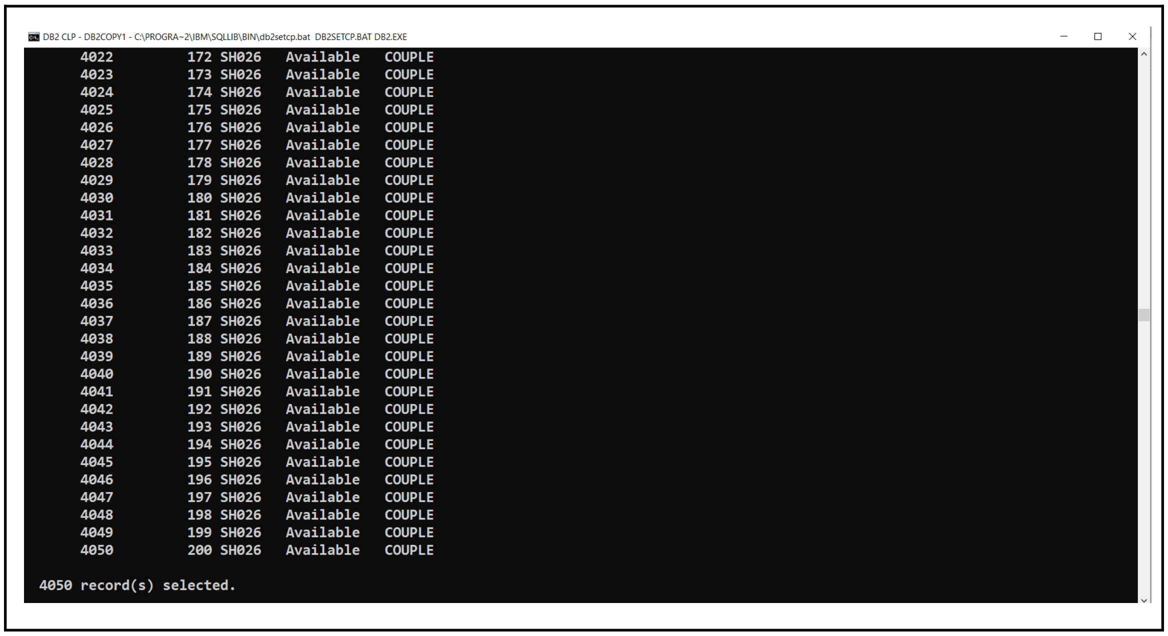
1) The purpose of this procedure is to save the time of administrator by auto generating multiple rows of seat ids for a show based on the parameters given.
2) This procedure generates and inserts multiple rows into the "seat" table based on the provided parameters.
3) The procedure required some input parameters which will be given in a trigger called trgShowtime.
   i) Show_ID
   ii) Hall_ID
   iii) Seat_ID
   iv) Sequence
   v) Seat_Status
   vi) Seat_Type
4) It retrieves the total number of seats (Seat_Count) from the "Hall" table for a specific "Hall_ID" using the cursor and uses it to determine the number of iterations for the insertion loop.
5) Each row inserted will have an auto-generated "Seat_ID" and will use the values provided for "Show_ID", "Seat_Status", and "Seat_Type".
6) The "Sequence" parameter is used to determine the initial sequence value, which increments for each iteration.

Procedure Codes:

```
Create or replace procedure auto_seat
(
IN Show_ID varchar(5),
Hall_ID char(1),
Seat_ID int,
Sequence int,
Seat_Status varchar(10),
Seat_Type varchar(10)
)
Begin

Declare Total_Seat int;
Declare cursor1 cursor for Select
Seat_Count from Hall Where
Hall.Hall_ID = auto_seat.Hall_ID;

Open cursor1;
Fetch from cursor1 into Total_Seat;
Close cursor1;

While Sequence <= Total_Seat
Do
Insert into seat values
(
DEFAULT,
sequence,
show_ID,
seat_status,
Seat_type
);

Set sequence = sequence +1;

End while;
End
```

DEMONSTRATION USAGE OF PROCEDURE

```
Insert into showtime values
('SH001','M300','C','2023-06-15','19:30:00'),
('SH002','M200','A','2023-06-15','21:00:00'),
('SH003','M300','B','2023-06-15','15:00:00'),
('SH004','M400','A','2023-06-15','14:00:00'),
('SH005','M500','A','2023-06-16','10:00:00'),
('SH006','M100','B','2023-06-16','20:00:00'),
('SH007','M400','B','2023-06-17','16:30:00'),
('SH008','M500','A','2023-06-17','13:00:00'),
('SH009','M500','B','2023-06-17','19:00:00'),
('SH010','M200','C','2023-06-18','21:00:00'),
('SH011','M100','B','2023-06-18','17:00:00'),
('SH012','M500','A','2023-06-18','10:00:00'),
('SH013', 'M500', 'C', '2023-06-25', '10:30:00'),
('SH014', 'M100', 'C', '2023-06-25', '09:00:00'),
('SH015', 'M400', 'A', '2023-06-30', '14:45:00'),
('SH016', 'M500', 'B', '2023-06-30', '17:00:00'),
('SH017', 'M300', 'B', '2023-07-01', '18:30:00'),
('SH018', 'M200', 'C', '2023-07-05', '12:00:00'),
('SH019', 'M300', 'A', '2023-07-05', '19:00:00'),
('SH020', 'M400', 'A', '2023-07-15', '11:30:00'),
('SH021', 'M500', 'C', '2023-07-15', '16:45:00'),
('SH022', 'M100', 'B', '2023-07-15', '15:30:00'),
('SH023', 'M100', 'B', '2023-07-26', '10:00:00'),
('SH024', 'M400', 'C', '2023-07-26', '09:30:00'),
('SH025', 'M500', 'A', '2023-07-31', '15:00:00'),
('SH026', 'M200', 'A', '2023-07-31', '14:00:00')
```



DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

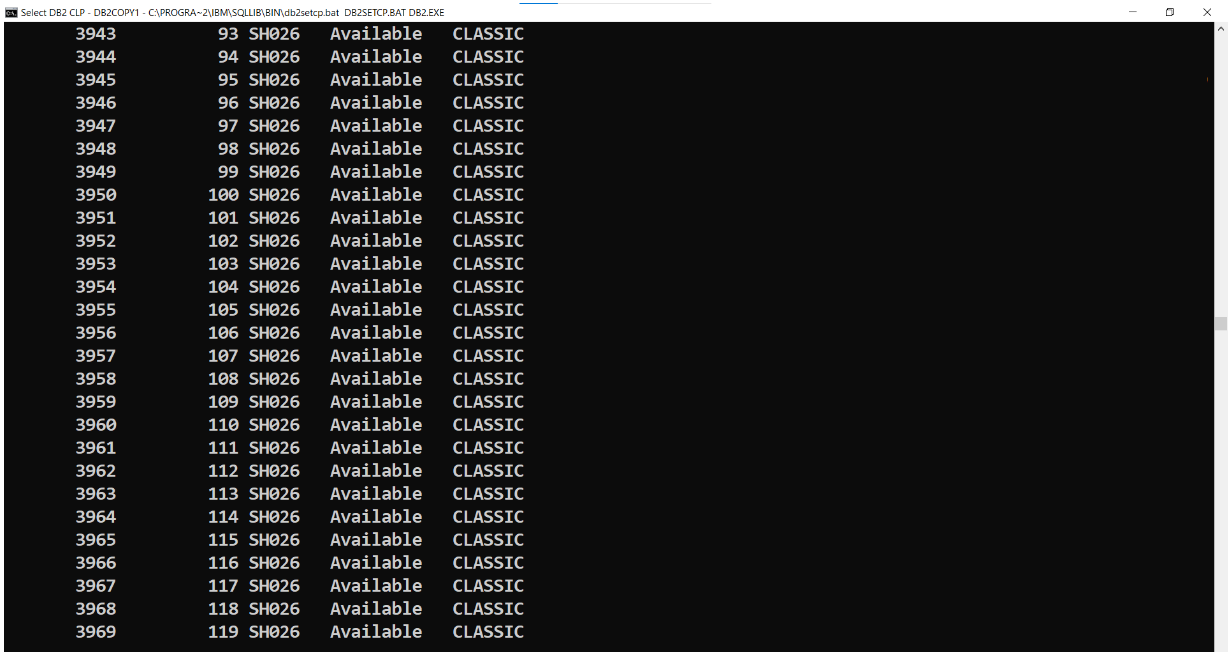
```
      4022         172 SH026   Available   COUPLE
      4023         173 SH026   Available   COUPLE
      4024         174 SH026   Available   COUPLE
      4025         175 SH026   Available   COUPLE
      4026         176 SH026   Available   COUPLE
      4027         177 SH026   Available   COUPLE
      4028         178 SH026   Available   COUPLE
      4029         179 SH026   Available   COUPLE
      4030         180 SH026   Available   COUPLE
      4031         181 SH026   Available   COUPLE
      4032         182 SH026   Available   COUPLE
      4033         183 SH026   Available   COUPLE
      4034         184 SH026   Available   COUPLE
      4035         185 SH026   Available   COUPLE
      4036         186 SH026   Available   COUPLE
      4037         187 SH026   Available   COUPLE
      4038         188 SH026   Available   COUPLE
      4039         189 SH026   Available   COUPLE
      4040         190 SH026   Available   COUPLE
      4041         191 SH026   Available   COUPLE
      4042         192 SH026   Available   COUPLE
      4043         193 SH026   Available   COUPLE
      4044         194 SH026   Available   COUPLE
      4045         195 SH026   Available   COUPLE
      4046         196 SH026   Available   COUPLE
      4047         197 SH026   Available   COUPLE
      4048         198 SH026   Available   COUPLE
      4049         199 SH026   Available   COUPLE
      4050         200 SH026   Available   COUPLE

  4050 record(s) selected.
```

☐ Triggers

Trigger Name:trgShowtime **Description**: 1)The purpose of this trigger is to automatically generate records into the Seat table when a new showtime is inserted without having the administrator to manually insert thousands of records. 2)This trigger is invoked after user inserts new record into showtime table. 3)It calls the auto_seat procedure to automatically generate seat records based on the inserted show_id and hall_id values. 4)The procedure is called with additional parameters such as the default value for Seat_ID, the initial value for Sequence, and default values for Seat_Status and Seat_Type.	**Trigger Codes:** Create or replace trigger trgShowtime After insert on showtime referencing new as n for each row mode db2sql Call auto_seat (n.show_id, n.hall_id, DEFAULT, 1, 'Available', 'CLASSIC')

DEMONSTRATION USAGE OF TRIGGER

```
Insert into showtime values
('SH001','M300','C','2023-06-15','19:30:00'),
('SH002','M200','A','2023-06-15','21:00:00'),
('SH003','M300','B','2023-06-15','15:00:00'),
('SH004','M400','A','2023-06-15','14:00:00'),
('SH005','M500','A','2023-06-16','10:00:00'),
('SH006','M100','B','2023-06-16','20:00:00'),
('SH007','M400','B','2023-06-17','16:30:00'),
('SH008','M500','A','2023-06-17','13:00:00'),
('SH009','M500','B','2023-06-17','19:00:00'),
('SH010','M200','C','2023-06-18','21:00:00'),
('SH011','M100','B','2023-06-18','17:00:00'),
('SH012','M500','A','2023-06-18','10:00:00'),
('SH013', 'M500', 'C', '2023-06-25', '10:30:00'),
('SH014', 'M100', 'C', '2023-06-25', '09:00:00'),
('SH015', 'M400', 'A', '2023-06-30', '14:45:00'),
('SH016', 'M500', 'B', '2023-06-30', '17:00:00'),
('SH017', 'M300', 'B', '2023-07-01', '18:30:00'),
('SH018', 'M200', 'C', '2023-07-05', '12:00:00'),
('SH019', 'M300', 'A', '2023-07-05', '19:00:00'),
('SH020', 'M400', 'A', '2023-07-15', '11:30:00'),
('SH021', 'M500', 'C', '2023-07-15', '16:45:00'),
('SH022', 'M100', 'B', '2023-07-15', '15:30:00'),
('SH023', 'M100', 'B', '2023-07-26', '10:00:00'),
('SH024', 'M400', 'C', '2023-07-26', '09:30:00'),
('SH025', 'M500', 'A', '2023-07-31', '15:00:00'),
('SH026', 'M200', 'A', '2023-07-31', '14:00:00')
```



Select DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
       3943          93 SH026    Available    CLASSIC
       3944          94 SH026    Available    CLASSIC
       3945          95 SH026    Available    CLASSIC
       3946          96 SH026    Available    CLASSIC
       3947          97 SH026    Available    CLASSIC
       3948          98 SH026    Available    CLASSIC
       3949          99 SH026    Available    CLASSIC
       3950         100 SH026    Available    CLASSIC
       3951         101 SH026    Available    CLASSIC
       3952         102 SH026    Available    CLASSIC
       3953         103 SH026    Available    CLASSIC
       3954         104 SH026    Available    CLASSIC
       3955         105 SH026    Available    CLASSIC
       3956         106 SH026    Available    CLASSIC
       3957         107 SH026    Available    CLASSIC
       3958         108 SH026    Available    CLASSIC
       3959         109 SH026    Available    CLASSIC
       3960         110 SH026    Available    CLASSIC
       3961         111 SH026    Available    CLASSIC
       3962         112 SH026    Available    CLASSIC
       3963         113 SH026    Available    CLASSIC
       3964         114 SH026    Available    CLASSIC
       3965         115 SH026    Available    CLASSIC
       3966         116 SH026    Available    CLASSIC
       3967         117 SH026    Available    CLASSIC
       3968         118 SH026    Available    CLASSIC
       3969         119 SH026    Available    CLASSIC
```

Trigger Name:trgSeatStatus
Description:
1)The purpose of this trigger is ensure that all seats that are inserted into ticket_info are changed from available to booked in the seat table to prevent customers from buying booked seats.

2)The trigger is created or replaced in DB2 and executed after an insertion on the "ticket_info" table. For each newly inserted row, the trigger updates the "seat_status" column of the corresponding row in the "seat" table, setting it to 'Booked.' The update is performed based on matching "seat_id" values between the "seat" and "ticket_info" tables.

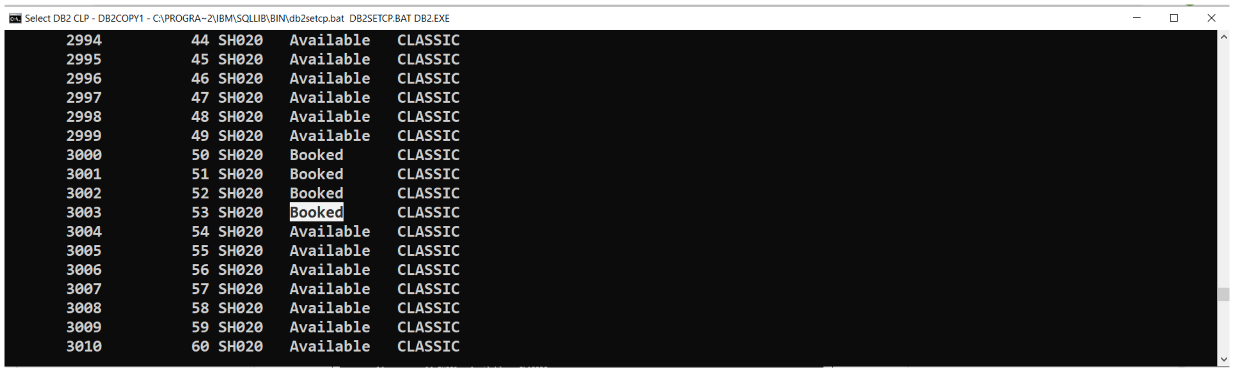
3)The trigger ensures that whenever a ticket is inserted into the "ticket_info" table, the corresponding seat's status in the "seat" table is updated to 'Booked.'

.

Trigger Codes:
```
Create or replace trigger trgSeatStatus
After insert on ticket_info
referencing new as n
for each row mode db2sql
Update seat
Set seat_status = 'Booked'
where n.seat_id = seat.seat_id
```

DEMONSTRATION USAGE OF TRIGGER

```
Insert into ticket_info values('K47',3003,'T15')
```


Select DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
      2994          44 SH020   Available   CLASSIC
      2995          45 SH020   Available   CLASSIC
      2996          46 SH020   Available   CLASSIC
      2997          47 SH020   Available   CLASSIC
      2998          48 SH020   Available   CLASSIC
      2999          49 SH020   Available   CLASSIC
      3000          50 SH020   Booked      CLASSIC
      3001          51 SH020   Booked      CLASSIC
      3002          52 SH020   Booked      CLASSIC
      3003          53 SH020   Booked      CLASSIC
      3004          54 SH020   Available   CLASSIC
      3005          55 SH020   Available   CLASSIC
      3006          56 SH020   Available   CLASSIC
      3007          57 SH020   Available   CLASSIC
      3008          58 SH020   Available   CLASSIC
      3009          59 SH020   Available   CLASSIC
      3010          60 SH020   Available   CLASSIC
```

Trigger Name:trgDropCustomer
Description:
1)The purpose of this trigger is to track deleted records from customer table by inserting it into RemovedCustomerInfo table when a customer did not bought any tickets.

2)The trigger code inserts the deleted customer records from the "Customer" table into the "RemovedCustomerInfo" table. It references the deleted rows as "oldCusRecord" using the referencing clause. The for each statement clause specifies that the trigger operates at the statement level.

3)Whenever a deletion occurs on the "Customer" table, this trigger captures the deleted customer records and inserts them into the "RemovedCustomerInfo" table, preserving the removed customer information for future reference or auditing purposes.

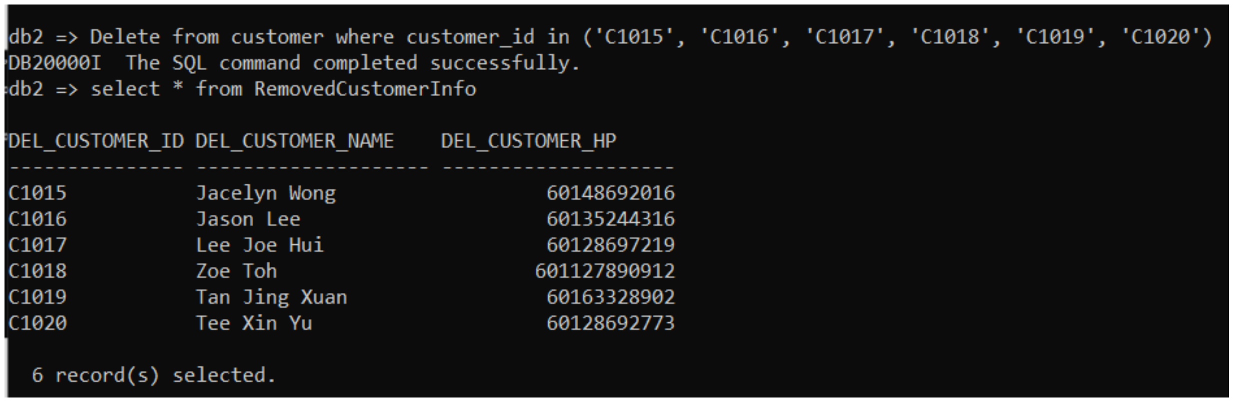
.

Trigger Codes:

```
Create or replace trigger trgDropCustomer
after delete on Customer
referencing old table as oldCusRecord
for each statement mode db2sql
insert into RemovedCustomerInfo
select * from oldCusRecord
```

DEMONSTRATION USAGE OF TRIGGER

```
Delete from customer where customer_id in ('C1015', 'C1016', 'C1017', 'C1018', 'C1019', 'C1020')
```



```
db2 => Delete from customer where customer_id in ('C1015', 'C1016', 'C1017', 'C1018', 'C1019', 'C1020')
DB20000I  The SQL command completed successfully.
db2 => select * from RemovedCustomerInfo

DEL_CUSTOMER_ID DEL_CUSTOMER_NAME    DEL_CUSTOMER_HP
--------------- -------------------- --------------------
C1015           Jacelyn Wong                  60148692016
C1016           Jason Lee                     60135244316
C1017           Lee Joe Hui                   60128697219
C1018           Zoe Toh                      601127890912
C1019           Tan Jing Xuan                 60163328902
C1020           Tee Xin Yu                    60128692773

  6 record(s) selected.
```

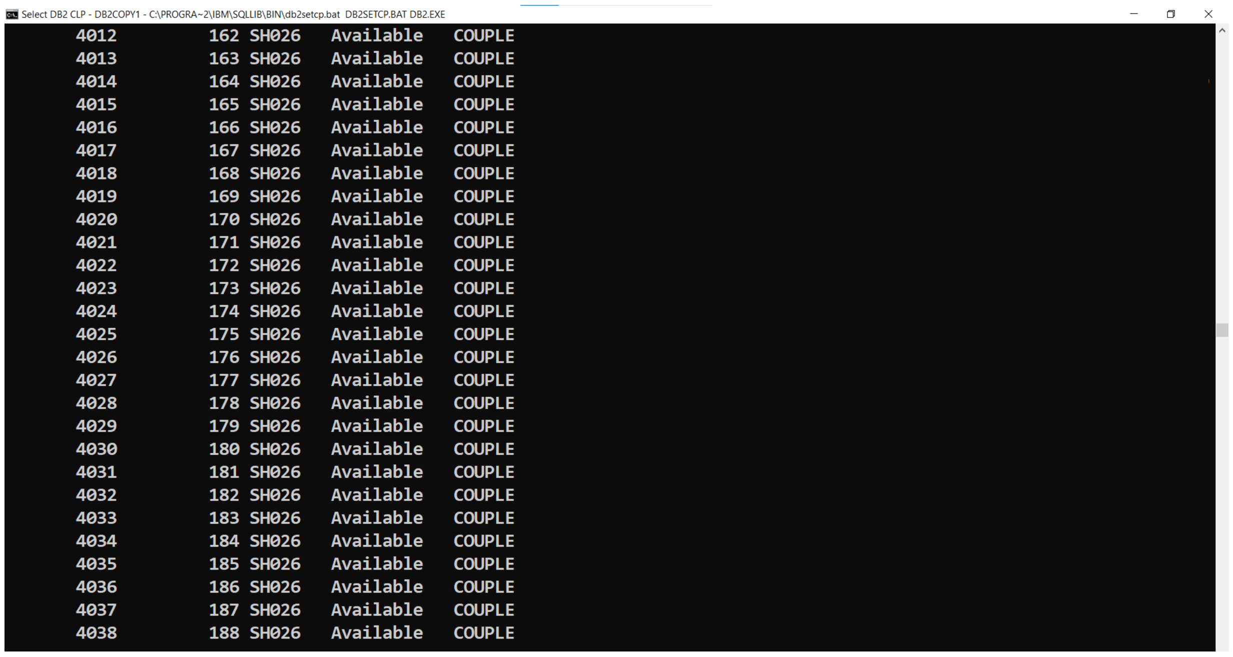
Trigger Name:trgSeatType
Description:
1)The purpose of this trigger is to update the last 20% of the hall's seats' seat type to couple seats.

2)The tigger will be called after each insertion in the seat table.

Trigger Codes:

```
Create or replace trigger trgSeatType
After insert on seat
for each row mode db2sql
Update seat
Set seat_type = 'COUPLE'
where sequence > 0.8*(select seat_count from
hall, showtime where showtime.show_id
=seat.show_id and
hall.hall_id=showtime.hall_id)
```

DEMONSTRATION USAGE OF TRIGGER
The trigger will automatically be called after trgShowtime

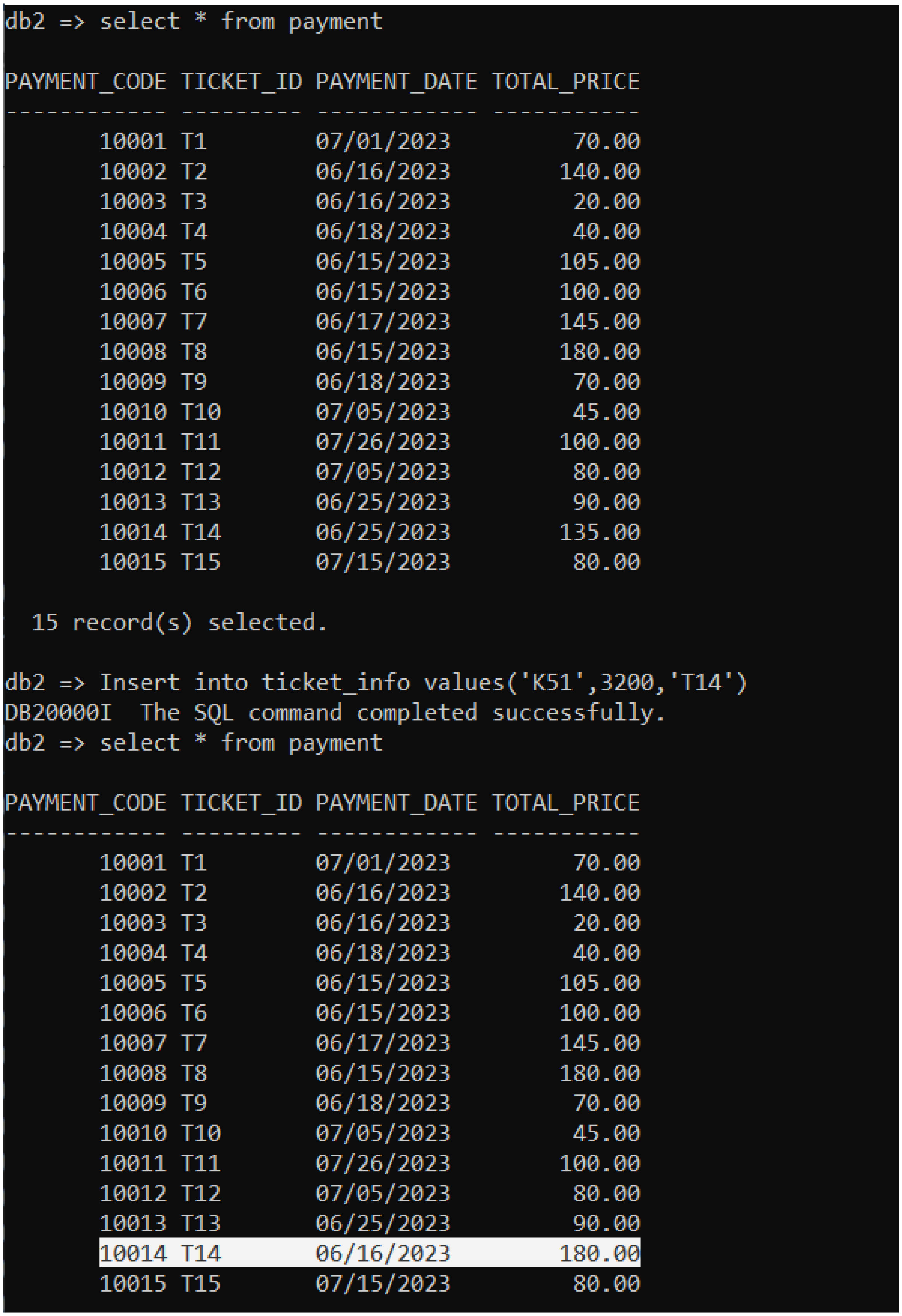


Select DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
        4012           162 SH026    Available    COUPLE
        4013           163 SH026    Available    COUPLE
        4014           164 SH026    Available    COUPLE
        4015           165 SH026    Available    COUPLE
        4016           166 SH026    Available    COUPLE
        4017           167 SH026    Available    COUPLE
        4018           168 SH026    Available    COUPLE
        4019           169 SH026    Available    COUPLE
        4020           170 SH026    Available    COUPLE
        4021           171 SH026    Available    COUPLE
        4022           172 SH026    Available    COUPLE
        4023           173 SH026    Available    COUPLE
        4024           174 SH026    Available    COUPLE
        4025           175 SH026    Available    COUPLE
        4026           176 SH026    Available    COUPLE
        4027           177 SH026    Available    COUPLE
        4028           178 SH026    Available    COUPLE
        4029           179 SH026    Available    COUPLE
        4030           180 SH026    Available    COUPLE
        4031           181 SH026    Available    COUPLE
        4032           182 SH026    Available    COUPLE
        4033           183 SH026    Available    COUPLE
        4034           184 SH026    Available    COUPLE
        4035           185 SH026    Available    COUPLE
        4036           186 SH026    Available    COUPLE
        4037           187 SH026    Available    COUPLE
        4038           188 SH026    Available    COUPLE
```

☐ Triggers With Subquery (Under Subquery)

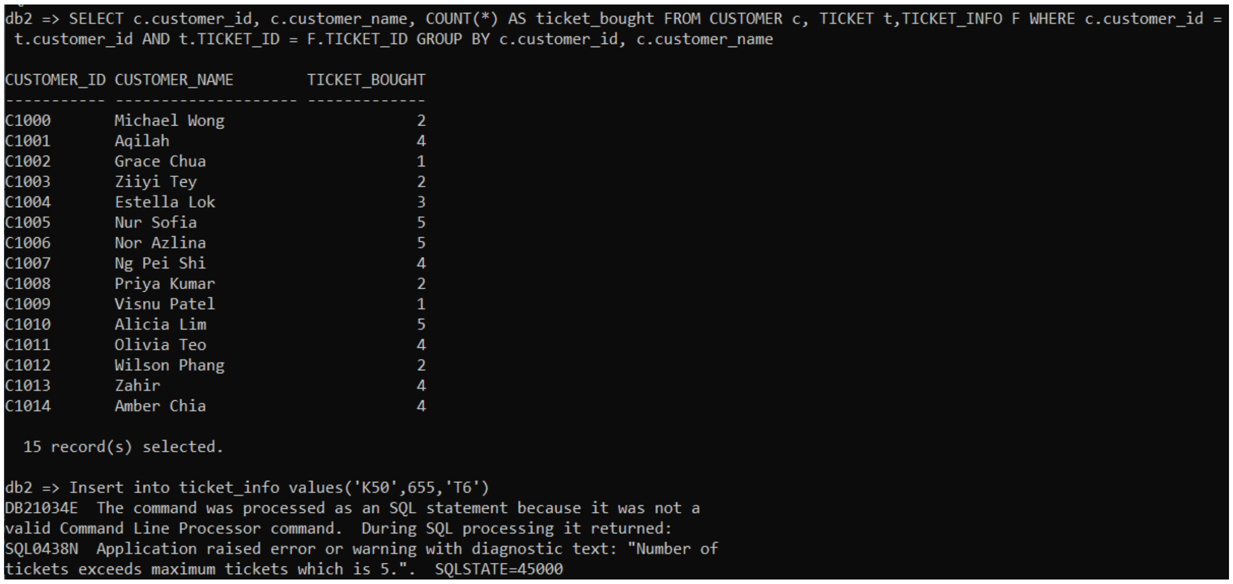
Trigger Name:trgtotalprice **Description**: 1)The trigger updates the total_price and payment_date columns in the payment table whenever a new row is inserted into the ticket_info table. 2)It calculates the new total_price by adding the pricing value associated with the seat of the newly inserted row by checking it's hall id and sets the payment_date to the current date for the corresponding ticket_id.	**Trigger Codes:** Create or replace trigger trgtotalprice After insert on ticket_info Referencing new as n for each row mode db2sql Begin Update payment set total_price = total_price + (SELECT pricing from customer as C, ticket as T, ticket_info as F, seat as S, showtime as Sh, hall as H where C.customer_id = T.customer_id and T.ticket_id = F.ticket_id and F.seat_id = S.seat_id and S.show_id = Sh.show_id and Sh.hall_id = H.hall_id and S.seat_id = n.seat_id) WHERE ticket_id = n.ticket_id; Update payment set payment_date = current date WHERE ticket_id = n.ticket_id; End
DEMONSTRATION USAGE OF TRIGGER Insert into ticket_info values('K51',3200,'T14')	



```
db2 => select * from payment

PAYMENT_CODE TICKET_ID PAYMENT_DATE TOTAL_PRICE
------------ --------- ------------ -----------
       10001 T1        07/01/2023         70.00
       10002 T2        06/16/2023        140.00
       10003 T3        06/16/2023         20.00
       10004 T4        06/18/2023         40.00
       10005 T5        06/15/2023        105.00
       10006 T6        06/15/2023        100.00
       10007 T7        06/17/2023        145.00
       10008 T8        06/15/2023        180.00
       10009 T9        06/18/2023         70.00
       10010 T10       07/05/2023         45.00
       10011 T11       07/26/2023        100.00
       10012 T12       07/05/2023         80.00
       10013 T13       06/25/2023         90.00
       10014 T14       06/25/2023        135.00
       10015 T15       07/15/2023         80.00

  15 record(s) selected.

db2 => Insert into ticket_info values('K51',3200,'T14')
DB20000I  The SQL command completed successfully.
db2 => select * from payment

PAYMENT_CODE TICKET_ID PAYMENT_DATE TOTAL_PRICE
------------ --------- ------------ -----------
       10001 T1        07/01/2023         70.00
       10002 T2        06/16/2023        140.00
       10003 T3        06/16/2023         20.00
       10004 T4        06/18/2023         40.00
       10005 T5        06/15/2023        105.00
       10006 T6        06/15/2023        100.00
       10007 T7        06/17/2023        145.00
       10008 T8        06/15/2023        180.00
       10009 T9        06/18/2023         70.00
       10010 T10       07/05/2023         45.00
       10011 T11       07/26/2023        100.00
       10012 T12       07/05/2023         80.00
       10013 T13       06/25/2023         90.00
       10014 T14       06/16/2023        180.00
       10015 T15       07/15/2023         80.00
```

☐ Triggers For Error Prompt (Under Queries Not Covered)

Trigger Name:check_ticket_sum **Description**: 1)This trigger calculates the number of tickets bought by a customer and ensures that the total count of tickets purchased by a customer does not exceed the specified threshold which is 5. 2)If the total count exceeds the threshold of 5, an exception is raised with the message "Number of tickets exceeds maximum tickets which is 5" and the customer will not be able to buy the new ticket.	**Trigger Codes:** Create or replace trigger check_ticket_sum After insert on ticket_info referencing new as n for each row mode db2sql begin declare total integer; declare threshold integer; Select count(*) into total from customer as C,ticket as T,ticket_info as F where C.customer_id = T.customer_id and T.ticket_id = F.ticket_id and F.ticket_id = n.ticket_id group by C.customer_id, C.customer_name; SET threshold = 5; IF total > threshold then signal SQLSTATE '45000' set message_text = 'Number of tickets exceeds maximum tickets which is 5.'; end if; end
DEMONSTRATION USAGE OF TRIGGER Insert into ticket_info values('K50',655,'T6')	

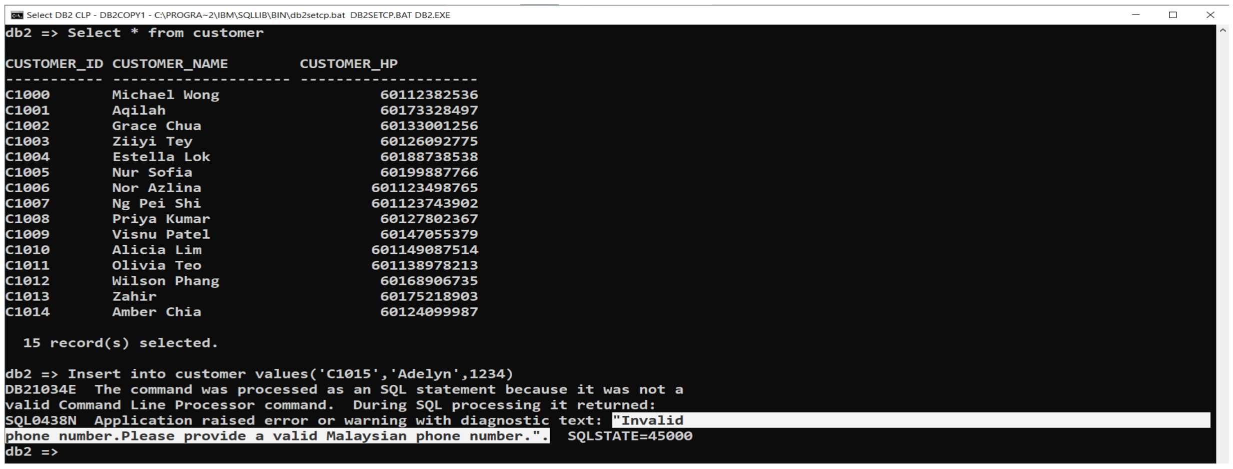


```
db2 => SELECT c.customer_id, c.customer_name, COUNT(*) AS ticket_bought FROM CUSTOMER c, TICKET t,TICKET_INFO F WHERE c.customer_id =
 t.customer_id AND t.TICKET_ID = F.TICKET_ID GROUP BY c.customer_id, c.customer_name

CUSTOMER_ID CUSTOMER_NAME        TICKET_BOUGHT
----------- -------------------- -------------
C1000       Michael Wong                     2
C1001       Aqilah                           4
C1002       Grace Chua                       1
C1003       Ziiyi Tey                        2
C1004       Estella Lok                      3
C1005       Nur Sofia                        5
C1006       Nor Azlina                       5
C1007       Ng Pei Shi                       4
C1008       Priya Kumar                      2
C1009       Visnu Patel                      1
C1010       Alicia Lim                       5
C1011       Olivia Teo                       4
C1012       Wilson Phang                     2
C1013       Zahir                            4
C1014       Amber Chia                       4

  15 record(s) selected.

db2 => Insert into ticket_info values('K50',655,'T6')
DB21034E  The command was processed as an SQL statement because it was not a
valid Command Line Processor command.  During SQL processing it returned:
SQL0438N  Application raised error or warning with diagnostic text: "Number of
tickets exceeds maximum tickets which is 5.".  SQLSTATE=45000
```

Trigger Name:trg_CustomerPhoneNumberValidation **Description**: 1)The trigger is designed to enforce the validation of valid Malaysian phone numbers for insertion into the "CUSTOMER" table. 2) If an invalid phone number is detected during the trigger execution, it raises an exception with the error message 'Invalid phone number. Please provide a valid Malaysian phone number.' 3)This ensures that only valid Malaysian phone numbers are allowed in the "CUSTOMER" table, and any attempt to insert an invalid phone number will result in an exception being raised with the specified error message.	**Trigger Codes:** Create trigger trg_CustomerPhoneNumberValidation Before insert on customer referencing new as n for each row mode db2sql begin declare exit handler for sqlstate '45000' begin signal sqlstate '45000' set message_text = 'Invalid phone number.Please provide a valid Malaysian phone number.'; end; if length(trim(n.customer_hp)) <> 11 and length(trim(n.customer_hp)) <> 12 then signal sqlstate '45000'; end if; end

DEMONSTRATION USAGE OF TRIGGER

Insert into customer values('C1015','Adelyn',1234)



Select DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => Select * from customer

CUSTOMER_ID CUSTOMER_NAME        CUSTOMER_HP
----------- -------------------- --------------------
C1000       Michael Wong                  60112382536
C1001       Aqilah                        60173328497
C1002       Grace Chua                    60133001256
C1003       Ziiyi Tey                     60126092775
C1004       Estella Lok                   60188738538
C1005       Nur Sofia                     60199887766
C1006       Nor Azlina                   601123498765
C1007       Ng Pei Shi                   601123743902
C1008       Priya Kumar                   60127802367
C1009       Visnu Patel                   60147055379
C1010       Alicia Lim                   601149087514
C1011       Olivia Teo                   601138978213
C1012       Wilson Phang                  60168906735
C1013       Zahir                         60175218903
C1014       Amber Chia                    60124099987

  15 record(s) selected.

db2 => Insert into customer values('C1015','Adelyn',1234)
DB21034E  The command was processed as an SQL statement because it was not a
valid Command Line Processor command.  During SQL processing it returned:
SQL0438N  Application raised error or warning with diagnostic text: "Invalid
phone number.Please provide a valid Malaysian phone number.".  SQLSTATE=45000
db2 =>
```

☐ View

View Name:allMovie

Description:

1)The view is created to summarize the movie's screening information for the company to monitor the ongoing movies easily

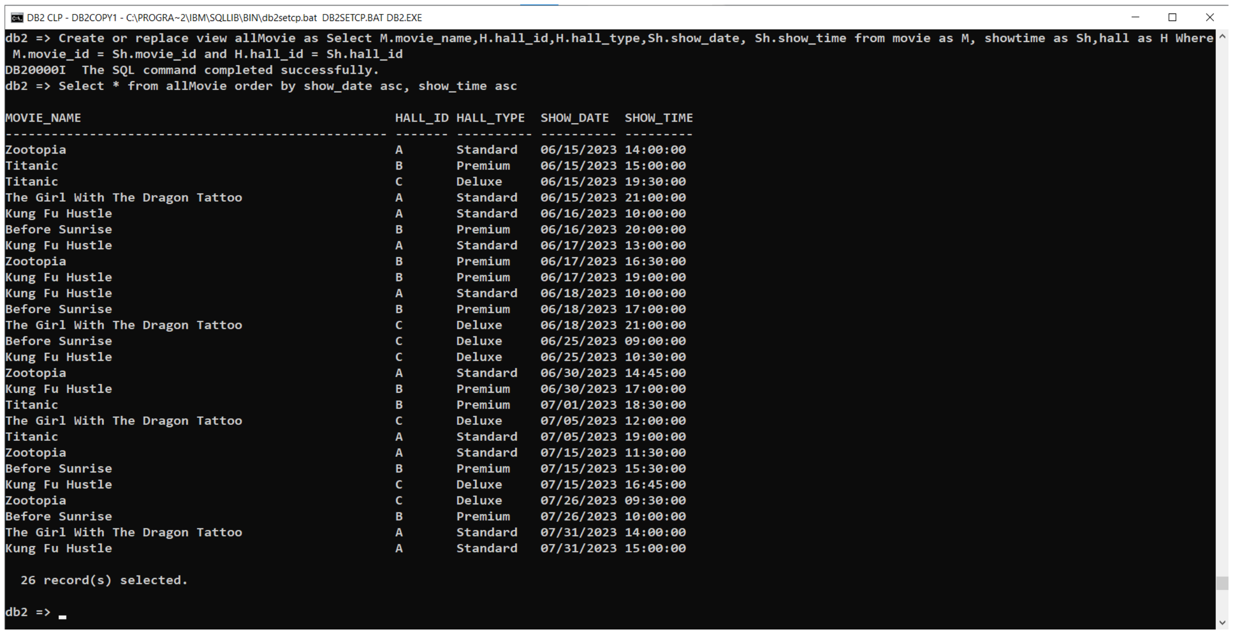
View Codes:

```
Create or replace view allMovie
as Select M.movie_name,
H.hall_id,
H.hall_type,
Sh.show_date,
Sh.show_time from movie as M,
showtime as Sh,
hall as H
Where M.movie_id = Sh.movie_id
and H.hall_id = Sh.hall_id
```

DEMONSTRATION USAGE OF VIEW

1)Displays the movie name, hall ID, hall type, show date, and show time. The records are sorted in ascending order based on the show date and show time.

```
Select * from allMovie order by show_date asc, show_time asc
```

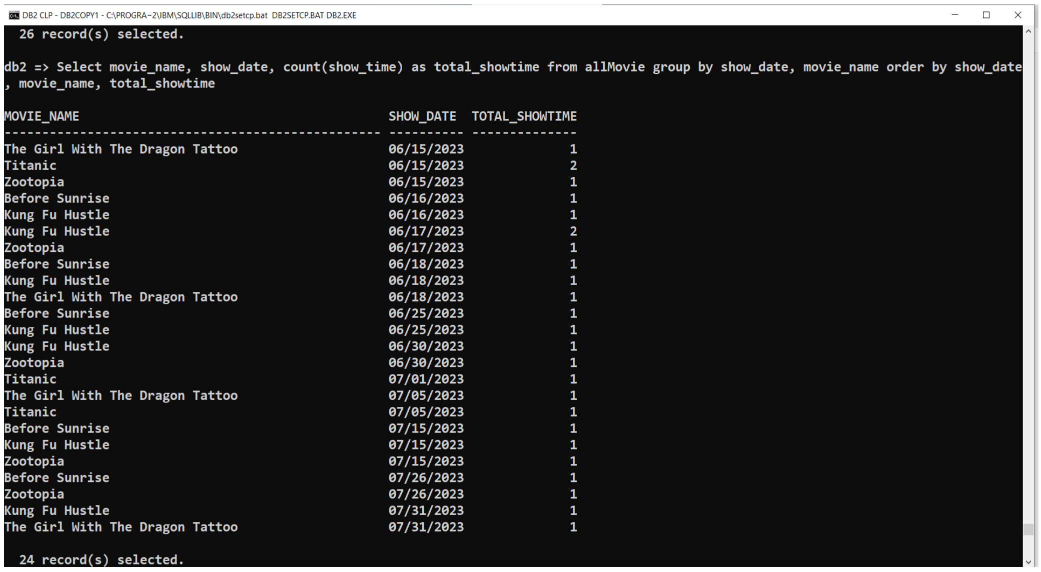


DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => Create or replace view allMovie as Select M.movie_name,H.hall_id,H.hall_type,Sh.show_date, Sh.show_time from movie as M, showtime as Sh,hall as H Where
 M.movie_id = Sh.movie_id and H.hall_id = Sh.hall_id
DB20000I  The SQL command completed successfully.
db2 => Select * from allMovie order by show_date asc, show_time asc

MOVIE_NAME                                         HALL_ID HALL_TYPE  SHOW_DATE  SHOW_TIME
-------------------------------------------------- ------- ---------- ---------- ---------
Zootopia                                           A       Standard   06/15/2023 14:00:00
Titanic                                            B       Premium    06/15/2023 15:00:00
Titanic                                            C       Deluxe     06/15/2023 19:30:00
The Girl With The Dragon Tattoo                    A       Standard   06/15/2023 21:00:00
Kung Fu Hustle                                     A       Standard   06/16/2023 10:00:00
Before Sunrise                                     B       Premium    06/16/2023 20:00:00
Kung Fu Hustle                                     A       Standard   06/17/2023 13:00:00
Zootopia                                           B       Premium    06/17/2023 16:30:00
Kung Fu Hustle                                     B       Premium    06/17/2023 19:00:00
Kung Fu Hustle                                     A       Standard   06/18/2023 10:00:00
Before Sunrise                                     B       Premium    06/18/2023 17:00:00
The Girl With The Dragon Tattoo                    C       Deluxe     06/18/2023 21:00:00
Before Sunrise                                     C       Deluxe     06/25/2023 09:00:00
Kung Fu Hustle                                     C       Deluxe     06/25/2023 10:30:00
Zootopia                                           A       Standard   06/30/2023 14:45:00
Kung Fu Hustle                                     B       Premium    06/30/2023 17:00:00
Titanic                                            B       Premium    07/01/2023 18:30:00
The Girl With The Dragon Tattoo                    C       Deluxe     07/05/2023 12:00:00
Titanic                                            A       Standard   07/05/2023 19:00:00
Zootopia                                           A       Standard   07/15/2023 11:30:00
Before Sunrise                                     B       Premium    07/15/2023 15:30:00
Kung Fu Hustle                                     C       Deluxe     07/15/2023 16:45:00
Zootopia                                           C       Deluxe     07/26/2023 09:30:00
Before Sunrise                                     B       Premium    07/26/2023 10:00:00
The Girl With The Dragon Tattoo                    A       Standard   07/31/2023 14:00:00
Kung Fu Hustle                                     A       Standard   07/31/2023 15:00:00

  26 record(s) selected.

db2 => 
```

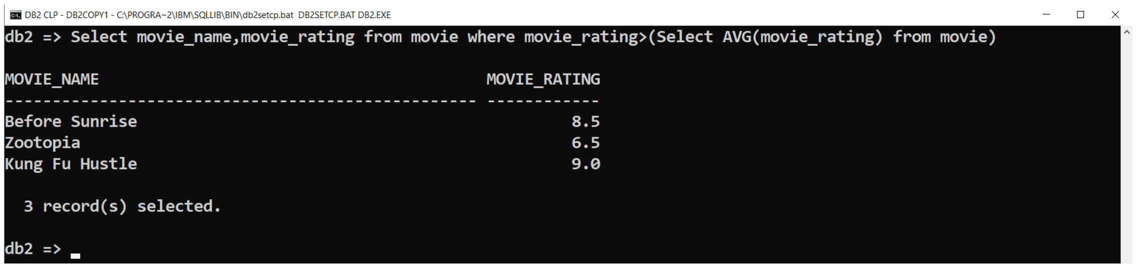
2)Count number of showtimes per day for each movie

Select movie_name, show_date, count(show_time) as total_showtime from allMovie group by show_date, movie_name order by show_date, movie_name, total_showtime

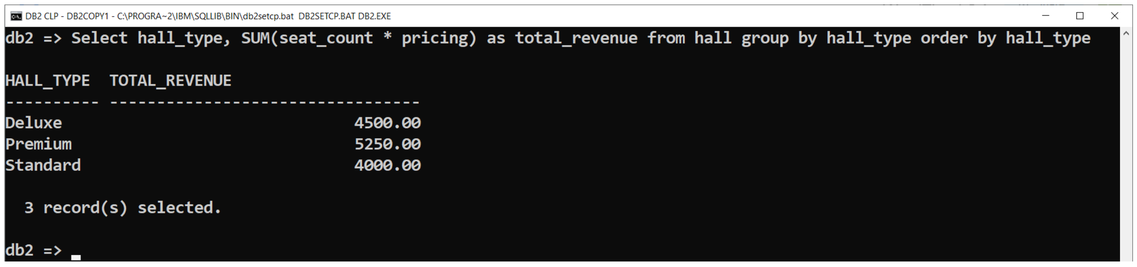

DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
  26 record(s) selected.

db2 => Select movie_name, show_date, count(show_time) as total_showtime from allMovie group by show_date, movie_name order by show_date
, movie_name, total_showtime

MOVIE_NAME                                         SHOW_DATE  TOTAL_SHOWTIME
-------------------------------------------------- ---------- --------------
The Girl With The Dragon Tattoo                    06/15/2023              1
Titanic                                            06/15/2023              2
Zootopia                                           06/15/2023              1
Before Sunrise                                     06/16/2023              1
Kung Fu Hustle                                     06/16/2023              1
Kung Fu Hustle                                     06/17/2023              2
Zootopia                                           06/17/2023              1
Before Sunrise                                     06/18/2023              1
Kung Fu Hustle                                     06/18/2023              1
The Girl With The Dragon Tattoo                    06/18/2023              1
Before Sunrise                                     06/25/2023              1
Kung Fu Hustle                                     06/25/2023              1
Kung Fu Hustle                                     06/30/2023              1
Zootopia                                           06/30/2023              1
Titanic                                            07/01/2023              1
The Girl With The Dragon Tattoo                    07/05/2023              1
Titanic                                            07/05/2023              1
Before Sunrise                                     07/15/2023              1
Kung Fu Hustle                                     07/15/2023              1
Zootopia                                           07/15/2023              1
Before Sunrise                                     07/26/2023              1
Zootopia                                           07/26/2023              1
Kung Fu Hustle                                     07/31/2023              1
The Girl With The Dragon Tattoo                    07/31/2023              1

  24 record(s) selected.
```

☐ Aggregate Function(Count,Max,Min,Avg,Sum)

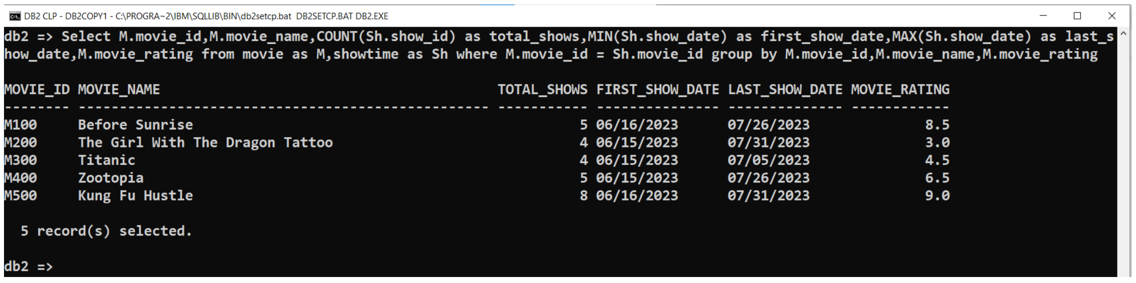
(AVG) **Description**: 1)Display movies with rating more than the average rating.	**Codes**: Select movie_name, movie_rating from movie where movie_rating>(Select AVG(movie_rating) from movie)
	

```
db2 => Select movie_name,movie_rating from movie where movie_rating>(Select AVG(movie_rating) from movie)

MOVIE_NAME                                         MOVIE_RATING
-------------------------------------------------- ------------
Before Sunrise                                              8.5
Zootopia                                                    6.5
Kung Fu Hustle                                              9.0

  3 record(s) selected.

db2 =>
```

(SUM) **Description**: 1)Calculates the total revenue for each hall type for administrator to see how much can they earn from a fully booked hall.	**Codes**: Select hall_type, SUM(seat_count * pricing) as total_revenue from hall group by hall_type order by hall_type
	

```
db2 => Select hall_type, SUM(seat_count * pricing) as total_revenue from hall group by hall_type order by hall_type

HALL_TYPE  TOTAL_REVENUE
---------- ---------------------------------
Deluxe                                4500.00
Premium                               5250.00
Standard                              4000.00

  3 record(s) selected.

db2 =>
```

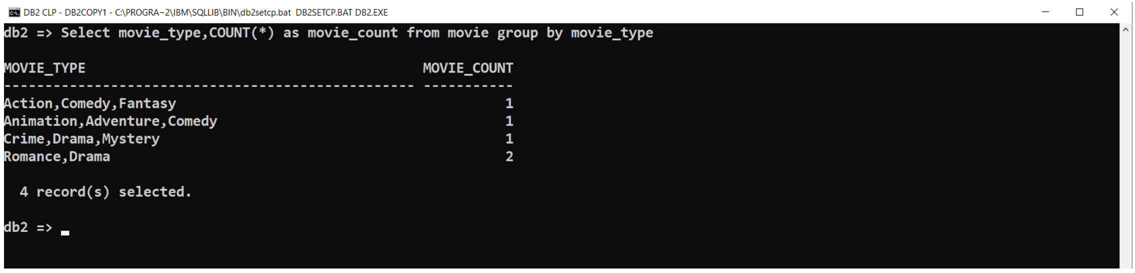
(MIN,MAX) **Description**: 1)Displays the movie_ID,movie_name,total numbers of shows,first show date,last show date ,and movie rating for each movie. 2)This allow customers to know how many showtimes are there for each movies and when will the movies stop airing.	**Codes**: Select M.movie_id, M.movie_name, COUNT(Sh.show_id) as total_shows, MIN(Sh.show_date) as first_show_date, MAX(Sh.show_date) as last_show_date, M.movie_rating from movie as M, showtime as Sh where M.movie_id = Sh.movie_id group by M.movie_id,M.movie_name,M.movie_rating



DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => Select M.movie_id,M.movie_name,COUNT(Sh.show_id) as total_shows,MIN(Sh.show_date) as first_show_date,MAX(Sh.show_date) as last_s
how_date,M.movie_rating from movie as M,showtime as Sh where M.movie_id = Sh.movie_id group by M.movie_id,M.movie_name,M.movie_rating

MOVIE_ID MOVIE_NAME                                         TOTAL_SHOWS FIRST_SHOW_DATE LAST_SHOW_DATE MOVIE_RATING
-------- -------------------------------------------------- ----------- --------------- -------------- ------------
M100     Before Sunrise                                               5 06/16/2023      07/26/2023              8.5
M200     The Girl With The Dragon Tattoo                              4 06/15/2023      07/31/2023              3.0
M300     Titanic                                                      4 06/15/2023      07/05/2023              4.5
M400     Zootopia                                                     5 06/15/2023      07/26/2023              6.5
M500     Kung Fu Hustle                                               8 06/16/2023      07/31/2023              9.0

  5 record(s) selected.

db2 =>
```

(COUNT) **Description**: 1)Calculates the total number of movies in each genre.	**Codes**: Select movie_type, COUNT(*) as movie_count from movie group by movie_type


DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => Select movie_type,COUNT(*) as movie_count from movie group by movie_type

MOVIE_TYPE                                         MOVIE_COUNT
-------------------------------------------------- -----------
Action,Comedy,Fantasy                                        1
Animation,Adventure,Comedy                                   1
Crime,Drama,Mystery                                          1
Romance,Drama                                                2

  4 record(s) selected.

db2 =>
```

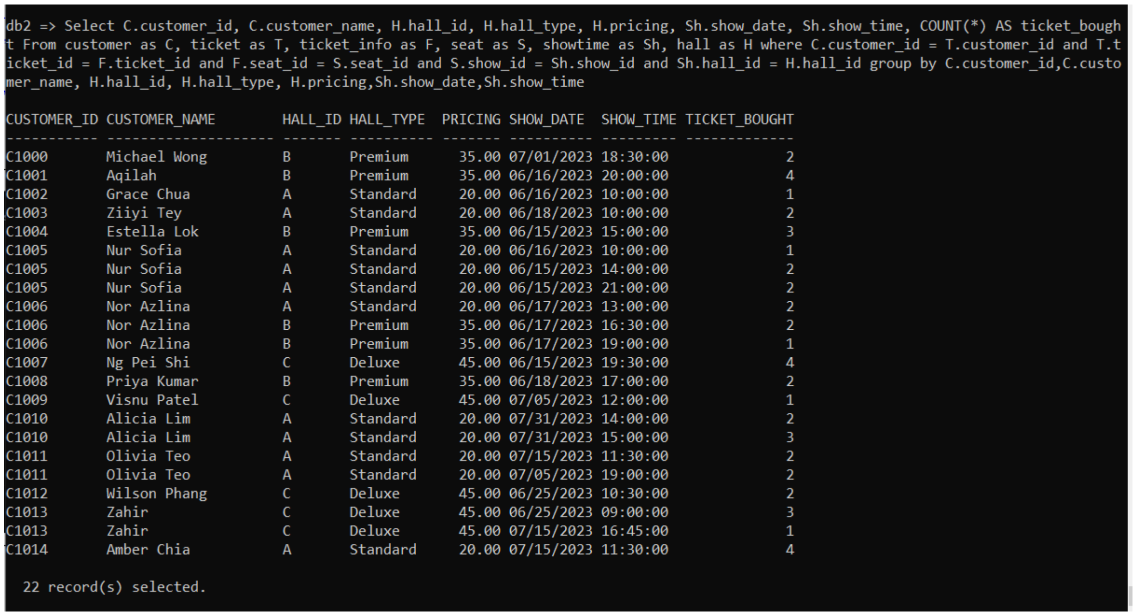
(COUNT)

Description:

1)This allows administrator to easily track and calculate the number of tickets bought by each customer along with the price for 1 ticket based on the hall type.

Codes:

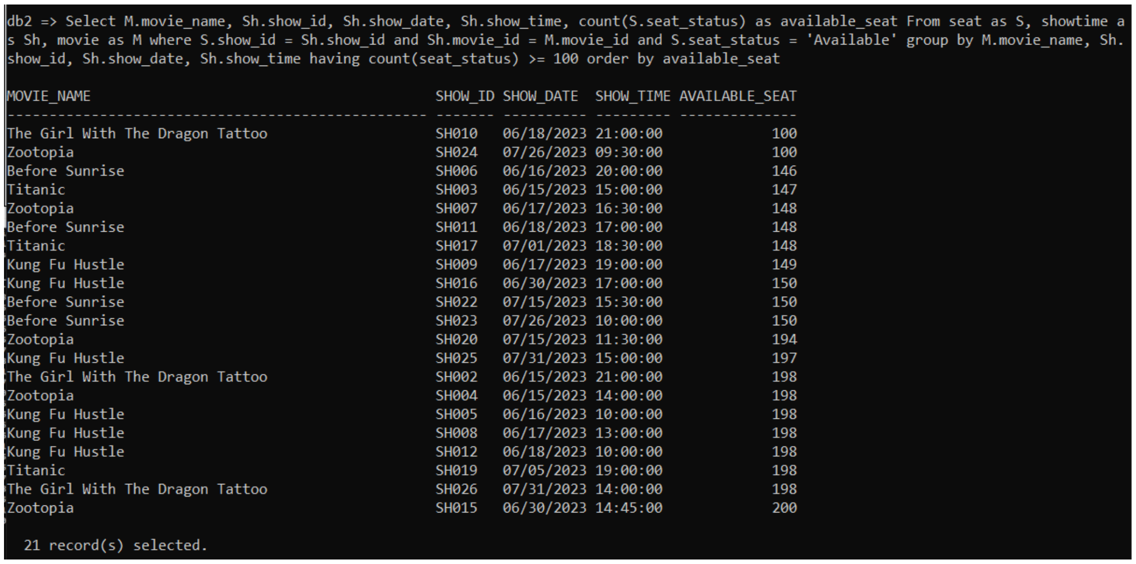
```
Select C.customer_id, C.customer_name,
H.hall_id,
H.hall_type,
H.pricing,
Sh.show_date,
Sh.show_time,
COUNT(*) AS ticket_bought
From customer as C, ticket as T, ticket_info as
F, seat as S, showtime as Sh, hall as H where
C.customer_id = T.customer_id
and T.ticket_id = F.ticket_id
and F.seat_id = S.seat_id
and S.show_id = Sh.show_id
and Sh.hall_id = H.hall_id
group by C.customer_id,C.customer_name,
H.hall_id, H.hall_type,
H.pricing,Sh.show_date,Sh.show_time
```



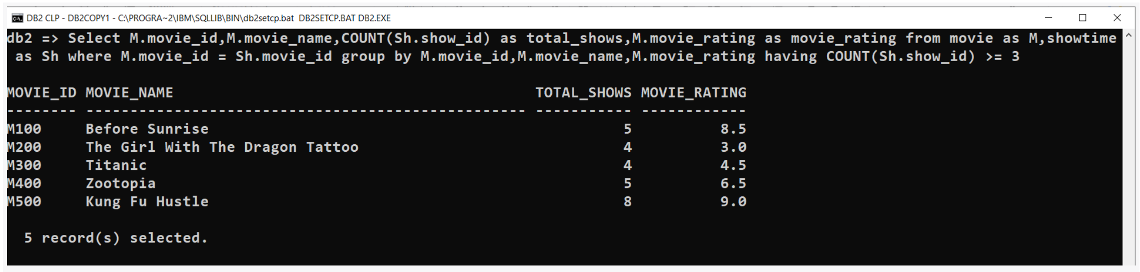
```
db2 => Select C.customer_id, C.customer_name, H.hall_id, H.hall_type, H.pricing, Sh.show_date, Sh.show_time, COUNT(*) AS ticket_bough
t From customer as C, ticket as T, ticket_info as F, seat as S, showtime as Sh, hall as H where C.customer_id = T.customer_id and T.t
icket_id = F.ticket_id and F.seat_id = S.seat_id and S.show_id = Sh.show_id and Sh.hall_id = H.hall_id group by C.customer_id,C.custo
mer_name, H.hall_id, H.hall_type, H.pricing,Sh.show_date,Sh.show_time

CUSTOMER_ID CUSTOMER_NAME        HALL_ID HALL_TYPE  PRICING SHOW_DATE  SHOW_TIME TICKET_BOUGHT
----------- -------------------- ------- ---------- ------- ---------- --------- -------------
C1000       Michael Wong         B       Premium      35.00 07/01/2023 18:30:00              2
C1001       Aqilah               B       Premium      35.00 06/16/2023 20:00:00              4
C1002       Grace Chua           A       Standard     20.00 06/16/2023 10:00:00              1
C1003       Ziiyi Tey            A       Standard     20.00 06/18/2023 10:00:00              2
C1004       Estella Lok          B       Premium      35.00 06/15/2023 15:00:00              3
C1005       Nur Sofia            A       Standard     20.00 06/16/2023 10:00:00              1
C1005       Nur Sofia            A       Standard     20.00 06/15/2023 14:00:00              2
C1005       Nur Sofia            A       Standard     20.00 06/15/2023 21:00:00              2
C1006       Nor Azlina           A       Standard     20.00 06/17/2023 13:00:00              2
C1006       Nor Azlina           B       Premium      35.00 06/17/2023 16:30:00              2
C1006       Nor Azlina           B       Premium      35.00 06/17/2023 19:00:00              1
C1007       Ng Pei Shi           C       Deluxe       45.00 06/15/2023 19:30:00              4
C1008       Priya Kumar          B       Premium      35.00 06/18/2023 17:00:00              2
C1009       Visnu Patel          C       Deluxe       45.00 07/05/2023 12:00:00              1
C1010       Alicia Lim           A       Standard     20.00 07/31/2023 14:00:00              2
C1010       Alicia Lim           A       Standard     20.00 07/31/2023 15:00:00              3
C1011       Olivia Teo           A       Standard     20.00 07/15/2023 11:30:00              2
C1011       Olivia Teo           A       Standard     20.00 07/05/2023 19:00:00              2
C1012       Wilson Phang         C       Deluxe       45.00 06/25/2023 10:30:00              2
C1013       Zahir                C       Deluxe       45.00 06/25/2023 09:00:00              3
C1013       Zahir                C       Deluxe       45.00 07/15/2023 16:45:00              1
C1014       Amber Chia           A       Standard     20.00 07/15/2023 11:30:00              4

  22 record(s) selected.
```

☐ Group By & Having Clauses

Description:	**Codes**:
1)Get movie name, showtime information (ID, date, time), and the count of available seats for each showtime. 2)The query filters the results to only include showtimes with at least 100 available seats. This allows administrators to see which showtime is not so popular among the customers.	Select M.movie_name, Sh.show_id, Sh.show_date, Sh.show_time, count(S.seat_status) as available_seat From seat as S, showtime as Sh, movie as M where S.show_id = Sh.show_id and Sh.movie_id = M.movie_id and S.seat_status = 'Available' group by M.movie_name, Sh.show_id, Sh.show_date, Sh.show_time having count(seat_status) >= 100 order by available_seat



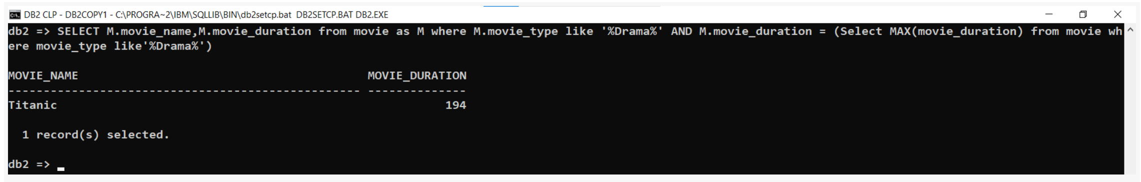
```
db2 => Select M.movie_name, Sh.show_id, Sh.show_date, Sh.show_time, count(S.seat_status) as available_seat From seat as S, showtime a
s Sh, movie as M where S.show_id = Sh.show_id and Sh.movie_id = M.movie_id and S.seat_status = 'Available' group by M.movie_name, Sh.
show_id, Sh.show_date, Sh.show_time having count(seat_status) >= 100 order by available_seat

MOVIE_NAME                                         SHOW_ID SHOW_DATE  SHOW_TIME AVAILABLE_SEAT
-------------------------------------------------- ------- ---------- --------- --------------
The Girl With The Dragon Tattoo                    SH010   06/18/2023 21:00:00             100
Zootopia                                           SH024   07/26/2023 09:30:00             100
Before Sunrise                                     SH006   06/16/2023 20:00:00             146
Titanic                                            SH003   06/15/2023 15:00:00             147
Zootopia                                           SH007   06/17/2023 16:30:00             148
Before Sunrise                                     SH011   06/18/2023 17:00:00             148
Titanic                                            SH017   07/01/2023 18:30:00             148
Kung Fu Hustle                                     SH009   06/17/2023 19:00:00             149
Kung Fu Hustle                                     SH016   06/30/2023 17:00:00             150
Before Sunrise                                     SH022   07/15/2023 15:30:00             150
Before Sunrise                                     SH023   07/26/2023 10:00:00             150
Zootopia                                           SH020   07/15/2023 11:30:00             194
Kung Fu Hustle                                     SH025   07/31/2023 15:00:00             197
The Girl With The Dragon Tattoo                    SH002   06/15/2023 21:00:00             198
Zootopia                                           SH004   06/15/2023 14:00:00             198
Kung Fu Hustle                                     SH005   06/16/2023 10:00:00             198
Kung Fu Hustle                                     SH008   06/17/2023 13:00:00             198
Kung Fu Hustle                                     SH012   06/18/2023 10:00:00             198
Titanic                                            SH019   07/05/2023 19:00:00             198
The Girl With The Dragon Tattoo                    SH026   07/31/2023 14:00:00             198
Zootopia                                           SH015   06/30/2023 14:45:00             200

  21 record(s) selected.
```

Description: 1)Provides a summary of movies that meet the criteria of having at least 3 shows, allowing customer to identify which movies have a significant number of showtimes.	**Codes**: Select M.movie_id, M.movie_name, COUNT(Sh.show_id) as total_shows, M.movie_rating as movie_rating from movie as M,showtime as Sh where M.movie_id = Sh.movie_id group by M.movie_id,M.movie_name,M.movie_rating having COUNT(Sh.show_id) >= 3
	

DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

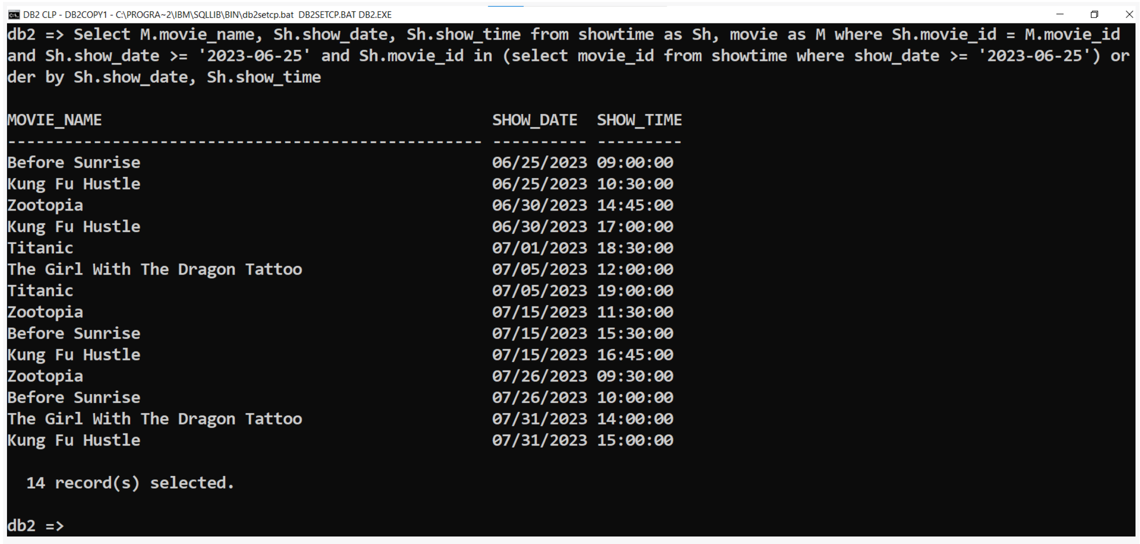
```
db2 => Select M.movie_id,M.movie_name,COUNT(Sh.show_id) as total_shows,M.movie_rating as movie_rating from movie as M,showtime
 as Sh where M.movie_id = Sh.movie_id group by M.movie_id,M.movie_name,M.movie_rating having COUNT(Sh.show_id) >= 3

MOVIE_ID MOVIE_NAME                                         TOTAL_SHOWS MOVIE_RATING
-------- -------------------------------------------------- ----------- ------------
M100     Before Sunrise                                               5          8.5
M200     The Girl With The Dragon Tattoo                              4          3.0
M300     Titanic                                                      4          4.5
M400     Zootopia                                                     5          6.5
M500     Kung Fu Hustle                                               8          9.0

  5 record(s) selected.
```

☐ Nested Queries / Subqueries

Description:	**Codes**:
1)Get the movie name and movie duration for movies that belong to the "Drama" genre and have the longest duration among all drama movies.	SELECT M.movie_name, M.movie_duration from movie as M where M.movie_type like '%Drama%' AND M.movie_duration = (Select MAX(movie_duration) from movie where movie_type like'%Drama%')
	

DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => SELECT M.movie_name,M.movie_duration from movie as M where M.movie_type like '%Drama%' AND M.movie_duration = (Select MAX(movie_duration) from movie wh
ere movie_type like'%Drama%')

MOVIE_NAME                                         MOVIE_DURATION
-------------------------------------------------- --------------
Titanic                                                       194

  1 record(s) selected.

db2 =>
```

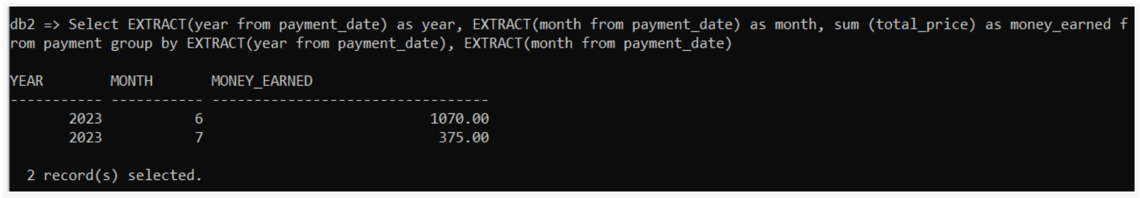
Description: 1)Display the movies that will be showing on June 25th 2023, and onwards.	**Codes**: Select M.movie_name, Sh.show_date, Sh.show_time from showtime as Sh, movie as M where Sh.movie_id = M.movie_id and Sh.show_date >= '2023-06-25' and Sh.movie_id in (select movie_id from showtime where show_date >= '2023-06-25') order by Sh.show_date, Sh.show_time
	

DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => Select M.movie_name, Sh.show_date, Sh.show_time from showtime as Sh, movie as M where Sh.movie_id = M.movie_id
and Sh.show_date >= '2023-06-25' and Sh.movie_id in (select movie_id from showtime where show_date >= '2023-06-25') or
der by Sh.show_date, Sh.show_time

MOVIE_NAME                                         SHOW_DATE  SHOW_TIME
-------------------------------------------------- ---------- ---------
Before Sunrise                                     06/25/2023 09:00:00
Kung Fu Hustle                                     06/25/2023 10:30:00
Zootopia                                           06/30/2023 14:45:00
Kung Fu Hustle                                     06/30/2023 17:00:00
Titanic                                            07/01/2023 18:30:00
The Girl With The Dragon Tattoo                    07/05/2023 12:00:00
Titanic                                            07/05/2023 19:00:00
Zootopia                                           07/15/2023 11:30:00
Before Sunrise                                     07/15/2023 15:30:00
Kung Fu Hustle                                     07/15/2023 16:45:00
Zootopia                                           07/26/2023 09:30:00
Before Sunrise                                     07/26/2023 10:00:00
The Girl With The Dragon Tattoo                    07/31/2023 14:00:00
Kung Fu Hustle                                     07/31/2023 15:00:00

  14 record(s) selected.

db2 =>
```

☐ To Check Money Earned In Year 2023 During June And July

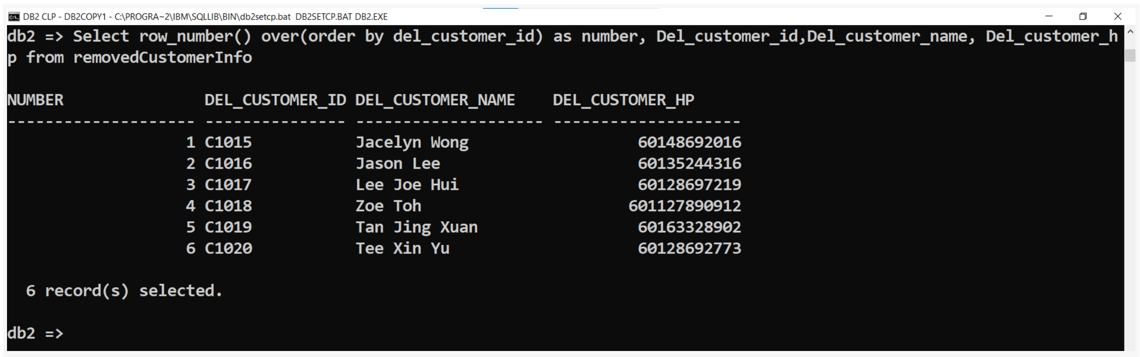
Description:	**Codes**:
1)Get the year, month, and total earnings from the "payment" table, grouped by year and month. 2)The purpose of this query is to see the total money earned throughout the year from each month by seeing how many seats are booked in each show.	Select EXTRACT(year from payment_date) as year, EXTRACT(month from payment_date) as month, sum (total_price) as money_earned from payment group by EXTRACT(year from payment_date), EXTRACT(month from payment_date)



```
db2 => Select EXTRACT(year from payment_date) as year, EXTRACT(month from payment_date) as month, sum (total_price) as money_earned f
rom payment group by EXTRACT(year from payment_date), EXTRACT(month from payment_date)

YEAR        MONTH       MONEY_EARNED
----------- ----------- ---------------------------------
       2023           6                           1070.00
       2023           7                            375.00

  2 record(s) selected.
```

☐ **Increment of How Many Removed Customer Info**

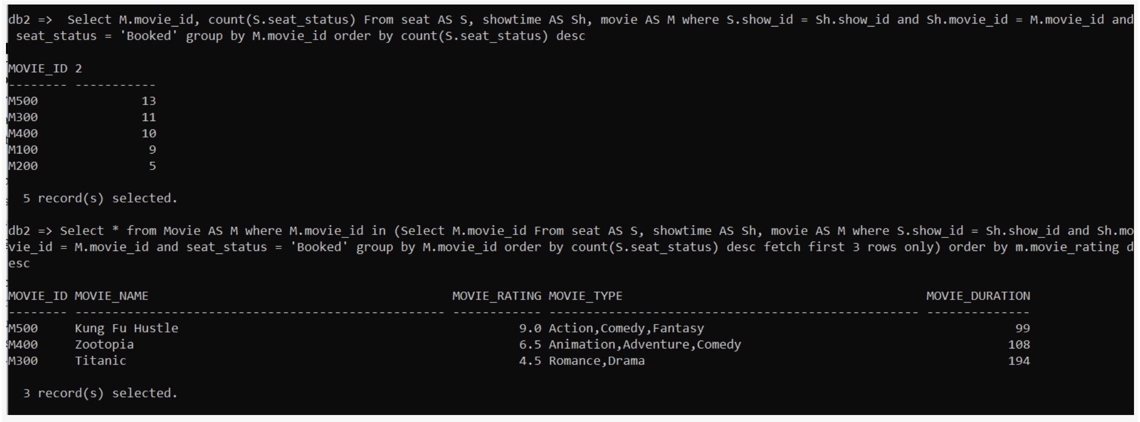
Description:	**Codes**:
1)The ROW_NUMBER() function is used to generate a sequential number for each record in the result set. The OVER(ORDER BY del_customer_id) clause specifies the ordering of the numbers based on the "del_customer_id" column. 2)The purpose of this query is to provide the removed customer information along with a sequential number assigned to each record. The "number" column represents the increment of how many removed customer records there are in the table. This allows administrator to easily see how many records have been removed.	Select row_number() over(order by del_customer_id) as number, Del_customer_id,Del_customer_name, Del_customer_hp from removedCustomerInfo


DB2 CLP - DB2COPY1 - C:\PROGRA~2\IBM\SQLLIB\BIN\db2setcp.bat DB2SETCP.BAT DB2.EXE

```
db2 => Select row_number() over(order by del_customer_id) as number, Del_customer_id,Del_customer_name, Del_customer_h
p from removedCustomerInfo

NUMBER               DEL_CUSTOMER_ID DEL_CUSTOMER_NAME    DEL_CUSTOMER_HP
-------------------- --------------- -------------------- --------------------
                   1 C1015           Jacelyn Wong                  60148692016
                   2 C1016           Jason Lee                     60135244316
                   3 C1017           Lee Joe Hui                   60128697219
                   4 C1018           Zoe Toh                      601127890912
                   5 C1019           Tan Jing Xuan                 60163328902
                   6 C1020           Tee Xin Yu                    60128692773

  6 record(s) selected.

db2 =>
```

☐ Display Top 3 Popular Movies Based On Booked Seat

Description:	**Codes**:
1)Get movie information from the "Movie" table based on movies that have the highest number of booked seats. It selects the top 3 movies with the highest counts of booked seats, based on grouping the records by movie ID and ordering them by the count of booked seats in descending order. 2)The purpose of this query is to display the top 3 movies in the cinema based on the number of seats booked by customers. This shows that these movies are commonly watched by customers in the cinema.	Select * from Movie AS M where M.movie_id in (Select M.movie_id From seat AS S, showtime AS Sh, movie AS M where S.show_id = Sh.show_id and Sh.movie_id = M.movie_id and seat_status = 'Booked' group by M.movie_id order by count(S.seat_status) desc fetch first 3 rows only) order by m.movie_rating desc



```
db2 =>  Select M.movie_id, count(S.seat_status) From seat AS S, showtime AS Sh, movie AS M where S.show_id = Sh.show_id and Sh.movie_id = M.movie_id and
 seat_status = 'Booked' group by M.movie_id order by count(S.seat_status) desc

MOVIE_ID 2
-------- -----------
M500              13
M300              11
M400              10
M100               9
M200               5

  5 record(s) selected.

db2 => Select * from Movie AS M where M.movie_id in (Select M.movie_id From seat AS S, showtime AS Sh, movie AS M where S.show_id = Sh.show_id and Sh.mo
vie_id = M.movie_id and seat_status = 'Booked' group by M.movie_id order by count(S.seat_status) desc fetch first 3 rows only) order by m.movie_rating d
esc

MOVIE_ID MOVIE_NAME                                         MOVIE_RATING MOVIE_TYPE                                         MOVIE_DURATION
-------- -------------------------------------------------- ------------ -------------------------------------------------- --------------
M500     Kung Fu Hustle                                              9.0 Action,Comedy,Fantasy                                          99
M400     Zootopia                                                    6.5 Animation,Adventure,Comedy                                    108
M300     Titanic                                                     4.5 Romance,Drama                                                 194

  3 record(s) selected.
```

☐ **Contributions**

1)Emily Phang Ru Ying	(1211102687)	100%
2)Teo Yu Jie	(1211102751)	100%
3)Lim Cai Qing	(1211102753)	100%