**Pre-Conditions Before Testing The Commands**

1. Create All Tables
2. Create all Trigger And Stored Procedures First
3. Insert movie->Hall->Customer->Showtime-> Select \*From Seat (auto insert) ->Ticket->Payment ->Ticket\_info
4. We have initially set the total\_price in payment as zero because it will be calculated automatically later by a trigger.
5. Alias:

Showtime = Sh

Seat=S

Hall=H

Movie=M

Ticket\_id = T

Ticket\_info=F

Customer = C

**CREATION OF TABLES**

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)

Create table hall(hall\_id char(1) not null primary key,hall\_type varchar(10),seat\_count int,pricing decimal(5,2))

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)

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)

Create table customer(customer\_id varchar(10) not null primary key,customer\_name varchar(20),customer\_hp bigint)

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

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)

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)

Create table RemovedCustomerInfo (Del\_customer\_id varchar(10), Del\_customer\_name varchar(20), Del\_customer\_hp bigint)

**STORED PROCEDURE**

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

**TRIGGERS**

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')

TO INVOKE:Insert into showtime values('SH026', 'M200', 'A', '2023-07-31', '14:00:00')

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  
TO INVOKE:Insert into ticket\_info values('K47',3003,'T15')

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

TO INVOKE:Delete from customer where customer\_id in ('C1015', 'C1016', 'C1017', 'C1018', 'C1019', 'C1020')

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)  
TO INVOKE:The trigger will automatically be called after trgShowtime

**Trigger With Subquery (Under Subquery)**

Before you use the trigger below, insert ticket\_info values using this trigger as we want to update all payment’s total\_price but at the same time we do not want all payments to be current date but only the new ones :

Create or replace trigger trgTotalPrice after insert on ticket\_info referencing new as n for each row mode db2sql update payment set total\_price = total\_price + (select pricing from CUSTOMER c, TICKET t,TICKET\_INFO F, seat s, showtime sh, hall 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

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

TO INVOKE:Insert into ticket\_info values('K51',3200,'T14')

TO SEE RESULT: Select \* from payment

**Error Prompt Trigger (Under Queries Not Covered)**

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

TO INVOKE: Insert into ticket\_info values('K50',655,'T6')

**Error Prompt Trigger (Under Queries Not Covered)**

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

TO INVOKE:Insert into customer values('C1015','Adelyn',1234)

**Data Insertion**

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)

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

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)  
  
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')

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')

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)

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')

**View**

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

Select \* from allMovie order by show\_date asc, show\_time asc

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

**Aggregate Function**

Select movie\_name,movie\_rating from movie where movie\_rating>(Select AVG(movie\_rating) from movie)

Select hall\_type, SUM(seat\_count \* pricing) as total\_revenue from hall group by hall\_type order by hall\_type

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

Select movie\_type,COUNT(\*) as movie\_count from movie group by movie\_type

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

**Group By And Having Clauses**

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

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

**Nested Queries / Subqueries**

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%')

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

**Queries Not Covered:**

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

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)

**Increment of How Many Removed Customer Info**

Select row\_number() over(order by del\_customer\_id) as number, Del\_customer\_id,Del\_customer\_name, Del\_customer\_hp from removedCustomerInfo

**Display Top 3 Popular Movies Based On Booked Seat**

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