WEEK 3 - MySQL

Day 1:

- create database service_station;
- use service_station;
- create table if not exists serviceStation (id int primary key,name varchar(10) ,address varchar(30) ,contact varchar(10));
- create table if not exists employees (id int primary key,name varchar(10) ,age int ,contact varchar(10));
- create table if not exists customers (id int primary key,name varchar(10) ,age int ,contact varchar(10), FOREIGN KEY (emp_id) REFERENCES employees(id));
- create table if not exists vehicles (id int primary key, type ENUM('Car','Bike','Bus'), brand varchar(10), color varchar(3), service_charge int);
- create table if not exists invoices (id int primary key, name_of_owner int REFERENCES customers(id), vehicle int REFERENCES vehicles(id) , amount_total int, employee_assigned int REFERENCES employees(id));
- drop table serviceStation , employees, customers, invoices, vehicles;
- drop database service_station;

Day 2 & 3:

- create database training_sample;
- use training_sample;

Loading the data:

- source sample-sqldata.sql

Queries that uses “students” table:

1. select * from students;
2. select name from students where name like 'H%';
3. select name from students where name like ‘%a%’;
4. select * from students order by name;
5. select * from students order by name LIMIT 2;
6. select * from students order by name LIMIT 2 OFFSET 2;

Queries that uses “marks” table:

1. select marks.id,student_id,subject_id,quarterly,half_yearly,annual,year,grade from students inner join marks on students.id = marks.student_id where annual IS NULL;

2. select student_id,subject_id,year from students inner join marks on students.id = marks.student_id where annual IS NULL && year = 2005;

3. select student_id ,subject_id,year from students inner join marks on students.id = marks.student_id where (annual IS NOT NULL && quarterly IS NULL && half_yearly IS NULL) || (annual IS NULL && quarterly IS NOT NULL && half_yearly IS NULL) || (annual IS NULL && quarterly IS NULL && half_yearly IS NOT NULL) ;

4.select student_id,subject_id,year,quarterly,half_yearly,annual from students inner join marks on students.id = marks.student_id where annual > 90 && quarterly > 90 && half_yearly > 90;

5. select student_id,subject_id,(IFNULL(quarterly,0)+IFNULL(half_yearly, 0)+IFNULL(annual,0))/3 as average,year from students inner join marks on students.id = marks.student_id group by subject_id,student_id,year;

6.select student_id,subject_id,(IFNULL(quarterly,0)+IFNULL(half_yearly, 0)+IFNULL(annual,0))/3 as average,year from students inner join marks on students.id = marks.student_id where year = 2003 || year = 2004 group by subject_id,student_id,year;

Advance Queries:

1. select name from students inner join marks on students.id = marks.student_id where annual IS NULL && quarterly IS NULL && half_yearly IS NULL;
2. select name, SUM(IFNULL(annual,0)) as marks ,year from students inner join marks on students.id = marks.student_id group by year,student_id;
3. select name , SUM(IFNULL(quarterly,0)) as total , grade from students inner join marks on students.id = marks.student_id where year = 2003 group by name;
4. select name,grade,COUNT(medal_won) as no_of_medal from students inner join medals on students.id = medals.student_id where grade = 9 || grade = 10 group by name having count(medal_won)>3 ;
5. select name,grade,COUNT(medal_won) as no_of_medal from students left join medals on students.id = medals.student_id group by name having count(medal_won) < 2 ;
6. select name,year from students inner join marks on students.id = marks.student_id where annual > 90 && name in (select name from students left join medals on students.id = medals.student_id where medals.id IS NULL) group by name,year having count(subject_id) = 5;
7. select name, game_id, medal_won, year, grade from students inner join medals on students.id = medals.student_id where student_id in (select student_id from medals group by student_id having count(medal_won) > 3) order by name;
8. - create table medal_temp (student_id int REFERENCES students(id), medals int, year int , grade int); - insert into medal_temp select student_id , count(medal_won) as medals, year , grade from medals group by student_id,grade;
   - select name, IFNULL(medals,0), SUM(quarterly)/count(*) as quarterly_per, SUM(half_yearly)/count(*) as half_yearly_per, SUM(annual)/count(*) as

annual_per, marks.year, marks.grade from students inner join marks on students.id = marks.student_id left join medal_temp on students.id = medal_temp.student_id group by name,year;

9. select name, calculate_grade(SUM(quarterly)) as quarterly_per, calculate_grade(SUM(half_yearly)) as half_yearly_per, calculate_grade(SUM(annual)) as annual_per, year, grade from students inner join marks on students.id = marks.student_id left

```
CREATE FUNCTION  calculate_grade(mark int) RETURNS VARCHAR(1)
DETERMINISTIC
BEGIN
   DECLARE g VARCHAR(1);

   IF mark < 200 THEN
 SET g = 'F';

 ELSEIF  mark < 250 THEN
       SET g = 'E';

ELSEIF  mark < 300 THEN
       SET g = 'D';


ELSEIF  mark < 350 THEN
       SET g = 'C';

ELSEIF  mark < 400 THEN
       SET g = 'B';

ELSEIF  mark < 450 THEN
       SET g = 'A';

ELSEIF  mark <= 500 THEN
       SET g = 'S';
END IF;

 RETURN (g);

END$$
```

Exercises:

- create table trains (TrainNo int(6) primary key, TrainName varchar(30));
- create table stations (StationId int(6) primary key, StationCode varchar(10));
- create table coaches (CoachCode int(6) primary key, CostPerKm float(5,2));

- create table users (UserId int(6) primary key, LoginId varchar(10), LPassword varchar(30));

- create table routes (RouteId int(6) primary key, OriginStationId int(6) REFERENCES stations(StationId), DestinationStationId int(6) REFERENCES stations(StationId) , DistanceInKms float(5,2));
- create table trainroutemaps (RouteId int(6) REFERENCES routes(RouteId),TrainNo int(6) REFERENCES trains(TrainNo), ArrivalTime TIME, DepartureTime TIME, DurationInMins int(2), primary key(RouteId,TrainNo));
- create table bookings (BookingRefNo int(6) primary key , RouteID int(6) REFERENCES routes(RouteId),TrainNo int(6) REFERENCES trains(TrainNo), CoachCode int (6) REFERENCES coaches(CoachCode),DateOfJourney date, DateofBooking date, NoOfTickets int);
- create table trainCoaches (TrainNo int(6) REFERENCES trains(TrainNo), CoachCode int (6) REFERENCES coaches(CoachCode), NoOfSeats int);

Queries:

1. select * from trains;
2. select RouteId, o.StationCode, d.StationCode , DistanceInKms from routes inner join stations o on (routes.OriginStationId = o.StationId) inner join stations d on (routes.DestinationStationId = d.StationId);
3. select TrainName , SUM(NoOfSeats) as NoOfSeats from trains inner join trainCoaches on trains.TrainNo = trainCoaches.TrainNo;
4. select RouteId, o.StationCode, d.StationCode , DistanceInKms from routes inner join stations o on (routes.OriginStationId = o.StationId) inner join stations d on (routes.DestinationStationId = d.StationId) where d.StationCode like ‘BGL’;
5. select RouteId, o.StationCode, d.StationCode , DistanceInKms from routes inner join stations o on (routes.OriginStationId = o.StationId) inner join stations d on (routes.DestinationStationId = d.StationId) where o.StationCode like ‘BGL’ ll o.StationCode like ‘CBE’ ll o.StationCode like ‘MAS’;
6. select BookingRefNo from bookings where DateOfBooking > '20050101' && DateOfBooking < '20050131';
7. select TrainName from trains where TrainName like ‘B%’;
8. select BookingRefNo from bookings where DateOfBooking IS NOT NULL;
9. select BookingRefNo from bookings where year(DateOfBooking) = '2006' && year(DateOfJourney) = '2007’;
10. select trains.TrainNo , TrainName , count(CoachCode) as NoOfCoaches from trains inner join trainCoaches on trains.TrainNo = trainCoaches.TrainNo group by TrainNo;
11. select count(*) as NoOfBookings from bookings where TrainNo = 16198;
12. select sum(NoOfTickets) as TicketsBooked from bookings where TrainNo = 14198;
13. select RouteId, o.StationCode, d.StationCode , DistanceInKms from routes inner join stations o on (routes.OriginStationId = o.StationId) inner join stations d on (routes.DestinationStationId = d.StationId) where (RouteId,DistanceInKms) = (select RouteId ,min(DistanceInKms) from routes);
14. select trains.TrainNo,trains.TrainName,sum(NoOfTickets) as TicketsBooked from bookings inner join trains on trains.TrainNo = bookings.TrainNo group by

TrainNo;

15. select CoachCode , (CostPerKm * 50) as CostPer50Km from coaches;
16. select TrainName , DepartureTime from trainroutemaps inner join trains on trains.TrainNo = trainroutemaps.TrainNo where RouteId IN (select RouteId from routes inner join stations on stations.StationId = routes.OriginStationId where StationCode like ‘BGL’);
17. select trains.TrainNo,trains.TrainName,sum(NoOfTickets) as TicketsBooked from bookings inner join trains on trains.TrainNo = bookings.TrainNo group by TrainNo having sum(NoOfTickets) > 500;
18. select trains.TrainNo,trains.TrainName,sum(NoOfTickets) as TicketsBooked from bookings inner join trains on trains.TrainNo = bookings.TrainNo group by TrainNo having sum(NoOfTickets) < 50;
19. select BookingRefNo, TrainName, trains.TrainNo , bookings.CoachCode , DateofJourney , NoOfTickets from bookings inner join trains on bookings.TrainNo = trains.TrainNo inner join routes r on routes.RouteId = bookings.RouteId inner join stations o on o.StationId = r.OriginStationId inner join stations on d.StationId = r.DestinationStationId where DateOfBooking > ‘20150225’;
20. select TrainName from trainroutemaps inner join trains on trains.TrainNo = trainroutemaps.TrainNo where RouteId ALL (select RouteId from routes inner join stations o on o.StationId = routes.OriginStationId inner join stations on d.StationId = routes.DestinationStationId where o.StationCode like 'MYS’ && d.StationCode like 'MAS');
21. select TrainName , trains.TrainNo from trains inner join bookings on trains.TrainNo = bookings.TrainNo group by bookings.TrainNo;

Day 4:

Alter table commands:

- alter table medals add column updated_at time, add column created_at;
- alter table marks add column updated_at time, add column created_at;
- alter table students add column updated_at time, add column created_at;
- update marks set quarterly=0 where quarterly IS NULL;
- update marks set half_yearly=0 where half_yearly IS NULL;
- update marks set annual=0 where annual IS NULL;
- alter table marks modify column quarterly int NOT NULL;
- alter table marks modify column half_yearly int NOT NULL;
- alter table marks modify column annual int NOT NULL;
- alter table medals modify column updated_at time default curtime(), modify column created_at default curtime();
- create trigger medal_update_trigger before update on marks for each row begin set new.updated_at=now(); end;

Exercise:

- insert into student_summary select t1.name,t1.student_id, annual, t1.year , medal_received from

(select name,student_id, annual , year from students inner join marks on students.id = marks.student_id) t1 inner join

```
(select name,student_id, count(medal_won) as medal_received , year  from students
inner join medals on students.id = medals.student_id group by name,year) t2 on
t1.year = t2.year && t1.student_id=t2.student_id;
        – create trigger insert_avg_marks
            before insert of quarterly ,half_yearly , annual on marks
              for each row
                  begin
                        set new.avg = (new.quarterly + new.half_yearly + new.annual)/
3;

                  end;
    — create trigger update_avg_marks
            before update of quarterly ,half_yearly , annual on marks
              for each row
                  begin
                        set new.avg = (new.quarterly + new.half_yearly + new.annual)/
3;

                  end;
     —   alter table medals add medal_received varchar(10);
     — create trigger medal_update before update on medals for each row  begin set
new.medal_received = new.medal_won;end $$
      — alter table medals drop medal_won;
```