-- 1. According to the given diagram create STUDENTS , ACTIVITIES and SCHEDULE tables. (PK – PRIMARY KEY, FK – FOREIGN KEY, \* - NOT NULL )

create table activities(

a\_id number,

constraint a\_id\_pk

primary key (a\_id),

a\_name varchar2(50) not null,

cost number not null

)

create table students(

s\_id number,

constraint s\_id\_pk

primary key (s\_id),

first\_name varchar2(50),

last\_name varchar2(50),

phone\_number varchar2(50),

email varchar2(50)

)

create table schedule(

s\_id number,

constraint s\_id\_fk

foreign key (s\_id)

references students(s\_id),

a\_id number,

constraint a\_id\_fk

foreign key (a\_id)

references activities(a\_id),

s\_date date

)

-- 2. Insert data into students table from employees table.

insert into students

select employee\_id, first\_name, last\_name, phone\_number, email

from hr.employees;

select \* from students;

-- 3. Change phone number to ‘\*\*\*’ for students with s\_id > 200.

update students set phone\_number = '\*\*\*' where s\_id > 200;

-- 4. Update first name and last names of students in Upper cases.

update students set first\_name = upper(first\_name), last\_name = upper(last\_name);

select \* from students;

-- 5. Based on the students table populated with the following data, update the email to 'DSA' for all records whose s\_id is greater than 150.

update students set email = 'DSA' where s\_id > 150;

select \* from students where s\_id > 140;

-- 6. Create PROGRAMMERS table using records from EMPLOYEES where job\_id contains ‘PROG’ substring

create table programmers as

(select \* from hr.employees where job\_id like '%PROG%');

select \* from programmers;

-- 7. Delete records from students table where s\_id is between 150 and 160.

delete students where s\_id between 150 and 160;

-- 8. a) Insert some date into SCHEDULE, then truncate and see results. b) Drop schedule table

-- a)

insert into students

values(1000, 'Zeki', 'Zekili', '050', 'gmail');

insert into activities

values(1000, 'Zekinin isi', 1000);

insert into schedule

values(1000, 1000, sysdate);

select \* from schedule sc left join students st

on sc.s\_id = st.s\_id

left join activities a

on a.a\_id = sc.a\_id;

-- b)

drop table schedule;

-- 9. For any date given, write a script to find: a) The first and the last days of the next year; b) The first and the last days of the next month; c) The first and the last days of the previous month.

-- a)

select ADD\_MONTHS(trunc(sysdate,'year'), 12), last\_day(ADD\_MONTHS(trunc(sysdate,'year'), 23)) from dual;

-- b)

select trunc(ADD\_MONTHS(sysdate, 1), 'month'), last\_day(ADD\_MONTHS(sysdate, 1)) from dual;

-- c)

select trunc(ADD\_MONTHS(sysdate, -1), 'month'), last\_day(ADD\_MONTHS(sysdate, -1)) from dual;

-- Create a table named “Participants” which consists of first\_name, last\_name and salary (have to more than 10000).

create table participants as

(select first\_name, last\_name, salary from hr.employees where salary > 10000);

select \* from participants;

create table participants2(

first\_name varchar2(50),

last\_name varchar2(50),

salary number check (salary > 10000)

) ;

insert into participants2

values('zeki', 'zekili', 1000000);

select \* from participants2;