

EXERCISE BOOK FOR DATABASE SYSTEMS LAB

This work was supported by the construction EFOP-[3.4.3-16-2016-00021](#). The project was supported by the European Union, co-financed by the European Social Fund.

Exercises for Class 1

1. Select all of the employees

```
select *  
from HR.EMPLOYEES;
```

2. Select the employees' first and last names.

```
select FIRST_NAME, LAST_NAME  
from HR.EMPLOYEES;
```

3. Select the employees' first names, last names, salaries, and the salaries increased by 1000.

```
select FIRST_NAME, LAST_NAME, SALARY, SALARY+1000  
from HR.EMPLOYEES;
```

4. Select the employees' first names, last names, salaries, and annual salaries.

```
select FIRST_NAME, LAST_NAME, SALARY as S, 12*SALARY ANNUAL_SALARY  
from HR.EMPLOYEES;
```

5. Select the employees' full names in one column

```
select FIRST_NAME || LAST_NAME as FULL_NAME  
from HR.EMPLOYEES;
```

```
select FIRST_NAME || ' ' || LAST_NAME as FULL_NAME  
from HR.EMPLOYEES;
```

6. Select the department ids.

```
SELECT DEPARTMENT_ID  
FROM HR.EMPLOYEES;
```

7. Select the department ids. Only the distinct values should be in the result set.

```
SELECT DISTINCT DEPARTMENT_ID  
FROM HR.EMPLOYEES;
```

- 8. Select the department and job ids. Only the distinct values should be in the result set.**

```
SELECT DISTINCT department_id, job_id  
FROM HR.EMPLOYEES;
```

Exercises for Class 2

- 1. Select every country from the COUNTRIES table.**

```
select *  
from OLYMPICS.O_COUNTRIES;
```

- 2. Select the names, the areas and the populations of each country.**

```
select name, area, population  
from OLYMPICS.O_COUNTRIES;
```

- 3. Select the names and the population densities of each country.**

```
select name, population/area as "Population_density"  
from OLYMPICS.O_COUNTRIES;
```

```
select name, population/area Population_density  
from OLYMPICS.O_COUNTRIES;
```

- 4. List the capital of Hungary**

```
select capital  
from OLYMPICS.O_COUNTRIES  
where name = 'Hungary';
```

- 5. List all European countries.**

```
select *  
from OLYMPICS.O_COUNTRIES  
where CONTINENT = 'Europe';
```

- 6. List the countries whose population is higher than 1000000.**

```
select *  
from OLYMPICS.O_COUNTRIES  
where POPULATION > 1000000;
```

**7. List the countries whose population is higher than 1000000 but less than 2000000.
([1000000,2000000[)**

```
select *  
from OLYMPICS.O_COUNTRIES  
where POPULATION > 1000000 and POPULATION < 2000000;
```

8. Between uses closed interval: [1000000,2000000]

```
select *  
from OLYMPICS.O_COUNTRIES  
where POPULATION between 1000000 and 2000000;
```

9. List the country names between Hungary and Germany.

Remark: *Between can be used for character sequences too!*

```
select name  
from OLYMPICS.O_COUNTRIES  
where name between 'Germany' and 'Hungary';
```

10. List the Asian and the European countries.

```
select *  
from OLYMPICS.O_COUNTRIES  
where CONTINENT = 'Europe' or CONTINENT = 'Asia';
```

Remark: *in operator takes a set as a parameter. In this case {Europe, Asia}.*

```
select *  
from OLYMPICS.O_COUNTRIES  
where CONTINENT in ('Europe','Asia');
```

11. List all countries except for Hungary

```
select *  
from OLYMPICS.O_COUNTRIES  
where name != 'Hungary';
```

12. List all countries except for Hungary, Germany and China

```
select *  
from OLYMPICS.O_COUNTRIES  
where name not in ('Hungary','China','Germany');
```

```
select *  
from OLYMPICS.O_COUNTRIES  
where name != 'Hungary' and name != 'China' and name != 'Germany';
```

Remark: *with De Morgan's law*

```
select *  
from OLYMPICS.O_COUNTRIES
```

where not(name = 'Hungary' or name = 'China' or name = 'Germany');

Remark: *NULL value can mean the following:*

- *unavailable*
- *unknown*
- *unassigned*
- *inapplicable*

13. List the countries that have no capital.

```
select *  
from OLYMPICS.O_COUNTRIES  
where capital is null;
```

14. List the countries that have a capital.

```
select *  
from OLYMPICS.O_COUNTRIES  
where capital is not null;
```

Remark:

The LIKE operator is used in a WHERE clause to search for a specified pattern in a column. There are two wildcards used in conjunction with the LIKE operator:

- *The percent sign represents zero, one, or multiple characters*
- *The underscore represents a single character*

15. Select the countries whose name starts with 'G'.

```
select *  
from OLYMPICS.O_COUNTRIES  
where name like 'G%';
```

16. Select the countries whose name has a 'g' in it.

```
select *  
from OLYMPICS.O_COUNTRIES  
where name like '%g%';
```

17. Select the countries whose name has at least 2 'a'-s in it.

```
select *  
from OLYMPICS.O_COUNTRIES  
where name like '%a%a%';
```

18. Select the countries whose name has a double 'r' in it.

```
select *  
from OLYMPICS.O_COUNTRIES  
where name like '%rr%';
```

19. Select the countries whose capital name starts with 'B' and ends with 'n'

```
select *  
from OLYMPICS.O_COUNTRIES  
where capital like 'B%n';
```

20. Select the countries whose capital name has 4 characters.

```
select *  
from OLYMPICS.O_COUNTRIES  
where capital like '____';
```

21. Select the countries whose name has an _ sign in it

```
select *  
from OLYMPICS.O_COUNTRIES  
where name like '%\_%' ESCAPE '\';
```

22. Select every athlete from the ATHLETES table.

```
select *  
from OLYMPICS.O_ATHLETES;
```

23. Select the athletes' names and birthplaces.

```
select name, birthplace  
from OLYMPICS.O_ATHLETES;
```

Exercises for Class 3

1. round function

```
select round(3.54), round(3.14,1), round(123.14,-2)  
from dual;
```

```
select round(3.54)  
from dual;
```

```
select *  
from dual;
```

```
select round(3.14,-1)  
from dual;
```

```
select round(153.14,-3)  
from dual;
```

2. Other numeric functions

```
select mod(3,2)
from dual;
```

```
select abs(-5), exp(2), power(2,4)
from dual;
```

```
select sin(90), cos(0), trunc(5.62), sqrt(9)
from dual;
```

3. char functions

```
select lower('database systems'), upper('database systems'), initcap('database systems'),
       concat('database','systems'), substr('Database',1,4), length('Database')
from dual;
```

4. date functions

```
select sysdate
from dual;
```

```
select name, birthdate
from OLYMPICS.O_ATHLETES;
```

5. List the names and birthdates of each athlete. Display the birthdate in the Hungarian date format

Remark: Hungarian date format: year.month.day

```
select name, to_char(birthdate,'YYYY.MM.DD') as "Birthday in Hungarian format"
from OLYMPICS.O_ATHLETES;
```

6. List the names and birthdates of each athlete who was born after 1980.01.01

```
select name, to_char(birthdate,'YYYY.MM.DD') as "Birthdate in Hungarian format"
from OLYMPICS.O_ATHLETES
where BIRTHDATE > to_date('1980.01.01','YYYY.MM.DD');
```

7. List the names and birthdate of each athlete who was born before 1980.01.01

```
select name, to_char(birthdate,'YYYY.MM.DD') as "Birthdate in Hungarian format"
from OLYMPICS.O_ATHLETES
where BIRTHDATE < to_date('1980.01.01','YYYY.MM.DD');
```

Remark: Order by sorts the result set by the given columns.

8. How old are the athletes?

```
select name, sysdate-birthdate Days
from OLYMPICS.o_athletes;
```

```
select name, trunc(sysdate-birthdate) Days
from OLYMPICS.o_athletes;
```

```
select name, trunc((sysdate-birthdate)/365) Age
from OLYMPICS.o_athletes;
```

```
select name, months_between(sysdate,birthdate) Months
from OLYMPICS.o_athletes;
```

```
select name, trunc(months_between(sysdate,birthdate)/12) Age
from OLYMPICS.o_athletes;
```

9. Select the date 2 months after the current date.

```
select add_months(sysdate,2)
from dual;
```

10. Select the date 2 months before the current date.

```
select add_months(sysdate,-2)
from dual;
```

11. List the names of the athletes in alphabetical order.

```
select name
from OLYMPICS.O_ATHLETES
order by name;
```

12. List the names of the athletes in reversed alphabetical order.

```
select name
from OLYMPICS.O_ATHLETES
order by name DESC;
```

13. List the names of the athletes in alphabetical order. If two athletes had a same name then sort them by birthplace.

```
select name, birthplace
from OLYMPICS.O_ATHLETES
order by name, birthplace;
```

Remark: *NVL function: if a column value is NULL then it replaces the NULL value with a specified value.*

14. List every athlete. If an athlete's birthplace is missing then display the word "Missing".

```
select name, nvl(birthplace,'Missing') Birthplace
from OLYMPICS.O_ATHLETES;
```

15. Select those countries whose name has no 'a' letter in it and whose population is greater than 10.000.000 or it is missing.

```
select *
from OLYMPICS.o_countries
where name not like '%a%' and (population > 10000000 or population is null);
```

16. Select those individual athletes who were born after 1990.02.01.

```
select *
from OLYMPICS.o_athletes
where team = 'no' and birthdate > to_date('1990.02.01','YYYY.MM.DD');
```

Exercises for Class 4

Remark:

- *Group functions: count, max, min, sum, avg*
- *Single row functions such as round, mod etc. take one row as an argument and return*
- *Group functions operate on sets of rows to give one result per group*

1. List the maximum and minimum value of the populations of the countries.

```
select min(population), max(population)
from OLYMPICS.O_COUNTRIES;
```

2. List the average population.

```
select avg(population)
from OLYMPICS.O_COUNTRIES;
```

3. Round it to the nearest integer.

```
select round(avg(population)) avg_population
from OLYMPICS.o_countries;
```

4. List the first country name in alphabetical order.

```
select min(name)
from OLYMPICS.O_COUNTRIES;
```

5. How many countries are there?

```
select count(*)
from OLYMPICS.O_COUNTRIES;
```

```
select count(name)
from OLYMPICS.O_COUNTRIES;
```

6. How many continents are there?

Wrong: this counts all rows that have a non-null continent value.

```
select count(continent)
from OLYMPICS.O_COUNTRIES;
```

Right: this count how many different continents are there.

```
select count(distinct continent)
from OLYMPICS.O_COUNTRIES;
```

How many cities have an athlete (How many cities are there from where at least one person participated in the Olympic Games)?

```
select count(distinct birthplace)
from OLYMPICS.O_ATHLETES;
```

7. How many athletes are there?

```
select count(*) as "num_of_athletes"
from OLYMPICS.O_ATHLETES;
```

Remark:

A_ID cannot be null (because it is the primary key). Therefore, the two statements are the same.

```
select count(a_id)
from OLYMPICS.o_athletes;
```

8. Select the minimum value of the birthdates.

```
select to_char(min(birthdate), 'YYYY.MM.DD')
from OLYMPICS.O_ATHLETES;
```

9. List the average population for each continent.

```
select continent, avg(population)
from OLYMPICS.O_COUNTRIES
group by continent;
```

10. Without the NULL value

```
select continent, avg(population)
from OLYMPICS.O_COUNTRIES
where continent is not null
group by continent;
```

11. List the maximum population for each continent.

```
select continent, max(population)
from OLYMPICS.O_COUNTRIES
group by continent;
```

12. Display the word 'Other' instead of NULL.

```
select nvl(continent,'Other'), max(population)
from OLYMPICS.O_COUNTRIES
group by continent;
```

13. How many countries are there in each continent?

```
select continent, count(*) as "num_of_countries"
from OLYMPICS.O_COUNTRIES
group by continent;
```

Remark: *There is no null-valued country name:*

```
select continent, count(name) as "num_of_countries"
from OLYMPICS.O_COUNTRIES
group by continent;
```

14. How many athletes are there for each city?

```
select birthplace, count(*)
from OLYMPICS.O_ATHLETES
group by birthplace;
```

15. List those cities that have more than 5 athletes in alphabetical order.

Remark: *In the where you cannot place group functions!*

```
select birthplace, count(*)
from OLYMPICS.O_ATHLETES
group by birthplace
having count(*) > 5
order by birthplace;
```

Remark: *Using the having clause the following steps will be applied:*

- rows are grouped
- group function is applied
- display only those rows that match the condition after the having

16. How many countries are there in each continent? List only those continents where this value is more than 20. Do not display the continents that have NULL value.

```
select continent, count(*)
from OLYMPICS.O_COUNTRIES
where continent is not null
group by CONTINENT
having count(*) > 20;
```

Remark: *group functions can be nested.*

```
select max(count(*))
from OLYMPICS.o_countries
group by continent;
```

Remark: *Having can be used without group by.*

```
select round(avg(population)) avg_pop
from OLYMPICS.o_countries
having min(population) != 0;
```

17. How many country names start with an 'A' or an 'E' letter?

```
select count(*)
from OLYMPICS.o_countries
where name like 'A%' or name like 'E%';
```

18. How many countries have higher population density than 100?

```
select count(*)
from OLYMPICS.o_countries
where population / area > 100 and area != 0;
```

For each continent list the number of countries that have a population higher than 10.000.000.

```
select continent, count(*)
from OLYMPICS.o_countries
where population > 10000000
group by continent;
```

Exercises for Class 5

Remark: *We use the join operator when we need data from multiple tables.*

1. List the athletes' names, birthplaces and the countries they come from.

```
select OLYMPICS.O_ATHLETES.name, OLYMPICS.O_ATHLETES.birthplace,
OLYMPICS.O_COUNTRIES.name
from OLYMPICS.O_COUNTRIES join OLYMPICS.O_ATHLETES
on (OLYMPICS.O_ATHLETES.country_id = OLYMPICS.O_COUNTRIES.c_id);
```

```
select a.name, a.birthplace, c.name
from OLYMPICS.O_COUNTRIES c join OLYMPICS.O_ATHLETES a
on (a.country_id = c.c_id);
```

```
select a.name, a.birthplace, c.name
from OLYMPICS.O_COUNTRIES c, OLYMPICS.O_ATHLETES a
where a.COUNTRY_ID = c.C_ID;
```

- 2. List the names of the countries and the numbers of medals (gold,silver,bronze) they earned.**

```
select c.NAME, mt.GOLD, mt.silver, mt.BRONZE
from OLYMPICS.O_COUNTRIES c, OLYMPICS.O_MEDAL_TABLE mt
where c.C_ID = mt.COUNTRY_ID;
```

- 3. List the names of those countries that earned at least 1 medal.**

```
select c.NAME, mt.GOLD, mt.silver, mt.BRONZE
from OLYMPICS.O_COUNTRIES c, OLYMPICS.O_MEDAL_TABLE mt
where c.C_ID = mt.COUNTRY_ID and
(mt.GOLD > 0 or mt.silver > 0 or mt.BRONZE > 0);
```

- 4. List the athletes' names the places they finished in, and the names of the sports events.**

```
select a.name, r.place, se.name
from OLYMPICS.O_ATHLETES a, OLYMPICS.O_RESULTS r,
     OLYMPICS.O_SPORT_EVENTS se
where a.A_ID = r.ATHLETE_ID and r.EVENT_ID = se.e_id;
```

- 5. List the athletes' names the places they finished in, and the names of the sports events. List only those athletes who finished before the 5th place.**

```
select a.name, r.place, se.name
from OLYMPICS.O_ATHLETES a, OLYMPICS.O_RESULTS r,
     OLYMPICS.O_SPORT_EVENTS se
where a.A_ID = r.ATHLETE_ID and r.EVENT_ID = se.e_id
and r.place < 5;
```

- 6. List those athletes who earned a medal.**

```
select a.name, r.place
from OLYMPICS.O_ATHLETES a, OLYMPICS.O_RESULTS r
where a.A_ID = r.ATHLETE_ID and r.place < 4;
```

- 7. Select the sport disciplines and the corresponding sport events**

```
select e.NAME, d.NAME
from OLYMPICS.O_SPORT_EVENTS e, OLYMPICS.O_SPORT_DISCIPLINES d
where e.DISCipline_ID = d.D_ID;
```

- 8. How many events are there for each discipline?**

```
select d.name, count(*)
from OLYMPICS.O_SPORT_EVENTS e, OLYMPICS.O_SPORT_DISCIPLINES d
where d.d_id = e.DISCipline_ID
```

```
group by d.NAME;
```

Remark: The select above does not count the NULL values.

```
select d.name, count(e.name)
from OLYMPICS.O_SPORT_EVENTS e, OLYMPICS.O_SPORT_DISCIPLINES d
where d.d_id = e.DISCipline_ID
group by d.NAME;
```

```
select d.name, count(distinct e.name)
from OLYMPICS.O_SPORT_EVENTS e, OLYMPICS.O_SPORT_DISCIPLINES d
where d.d_id = e.DISCipline_ID
group by d.NAME;
```

9. How many sport discipline are there?

```
select count(*)
from OLYMPICS.O_SPORT_DISCIPLINES;
```

10. List those European athletes who earned a medal.

```
select distinct a.name
from OLYMPICS.o_athletes a, OLYMPICS.o_results r, OLYMPICS.o_countries c
where a.a_id = r.athlete_id and r.place < 4 and c.c_id = a.country_id
and c.continent = 'Europe';
```

```
select distinct a.name
from OLYMPICS.o_athletes a join OLYMPICS.o_results r
on (a.a_id = r.athlete_id)
join OLYMPICS.o_countries c
on (c.c_id = a.country_id)
where r.place < 4 and c.continent = 'Europe';
```

11. List Michael Phelps' results and the names of the corresponding sportevents. The best result should come first.

```
select a.name, r.place, se.name
from OLYMPICS.o_athletes a, OLYMPICS.o_results r, OLYMPICS.o_sport_events se
where a.a_id = r.athlete_id and r.event_id = se.e_id
and a.name = 'Michael Phelps'
order by 2;
--order by r.place
```

12. For each country, select the number of individual athletes.

```
select c.name, count(*) num_of_athletes
from OLYMPICS.o_athletes a, OLYMPICS.o_countries c
where a.country_id = c.c_id and team = 'no'
group by c.name;
```

13. Display only those where this value is more than 1!

```
select c.name, count(*) num_of_athletes
from OLYMPICS.o_athletes a, OLYMPICS.o_countries c
where a.country_id = c.c_id and team = 'no'
group by c.name
having count(*) > 1;
```

14. The highest number of individual athletes should come first. If 2 of these values are the same, then sort them by the names of the countries in alphabetical order.

```
select c.name, count(*) num_of_athletes
from OLYMPICS.o_athletes a, OLYMPICS.o_countries c
where a.country_id = c.c_id and team = 'no'
group by c.name
having count(*) > 1
order by 2 DESC, c.name;
--order by num_of_athletes DESC, 1;
```

15. Select the number of European teams.

```
select count(*)
from OLYMPICS.o_athletes a, OLYMPICS.o_countries c
where a.country_id = c.c_id and continent = 'Europe'
and team = 'yes';
```

16. Select those athletes who earned a gold medal and come from Budapest.

```
select a.name
from OLYMPICS.o_athletes a, OLYMPICS.o_results r
where a.a_id = r.athlete_id and birthplace = 'Budapest'
and place = 1;
```

Exercises for Class 6

1. Select the athletes' names and birthplaces. List only those whose birthplace starts with a letter 'D'.

```
select name, birthplace
from OLYMPICS.O_ATHLETES
where birthplace like 'D%';
```

2. Select those individual athletes whose name is longer than 10 characters.

```
select name, team, length(name)
from OLYMPICS.O_ATHLETES
where length(name) > 10 and team = 'no';
```

3. How many Hungarian athletes are there?

```
select count(*)
from OLYMPICS.O_ATHLETES a, OLYMPICS.O_COUNTRIES c
where c.C_ID = a.COUNTRY_ID and c.name = 'Hungary';
```

4. List the athletes who are more than 30 years old. List only the names, birthdates (in Hungarian form), and the ages rounded to the nearest integer for each athlete.

```
select name, to_char(birthdate,'YYYY.MM.DD'),
       round((sysdate-birthdate)/365) Age
from OLYMPICS.O_ATHLETES
where round((sysdate-birthdate)/365) > 30;
```

```
select name, to_char(birthdate,'YYYY.MM.DD'),
       round(months_between(sysdate,BIRTHDATE)/12) as "Age"
from OLYMPICS.O_ATHLETES
where round(months_between(sysdate,BIRTHDATE)/12) > 30;
```

5. Select the average population of Europe.

```
select avg(population)
from OLYMPICS.O_COUNTRIES
where continent = 'Europe';
```

6. List the maximum population for each continent. Sort the rows by the maximum value.

```
select continent,max(population)
from OLYMPICS.O_COUNTRIES
group by continent
order by max(population); --order by 2
```

7. Select the number of athletes for each country. List those countries that have more than 5 athletes.

```
select c.name, count(*)
from OLYMPICS.O_ATHLETES a, OLYMPICS.O_COUNTRIES c
where a.country_id = c.c_id
group by c.name
having count(*) > 5;
```

8. Select the employees' names and the departments they work in.

```
select e.FIRST_NAME, e.LAST_NAME, d.DEPARTMENT_NAME
from HR.DEPARTMENTS d, HR.EMPLOYEES e
where d.DEPARTMENT_ID = e.DEPARTMENT_ID;
```

```
select e.FIRST_NAME, e.LAST_NAME, d.DEPARTMENT_NAME
from HR.DEPARTMENTS d join HR.EMPLOYEES e
on (d.DEPARTMENT_ID = e.DEPARTMENT_ID);
```

```
select e.FIRST_NAME, e.LAST_NAME, d.DEPARTMENT_NAME
from HR.DEPARTMENTS d join HR.EMPLOYEES e
using (DEPARTMENT_ID);
```

9. Select the employees' names and the departments they work in. List those employees who have no dept. id too!

```
select e.FIRST_NAME, e.LAST_NAME, d.DEPARTMENT_NAME
from HR.DEPARTMENTS d right outer join HR.EMPLOYEES e
on (d.DEPARTMENT_ID = e.DEPARTMENT_ID);
```

10. Select the employees' names and the departments they work in. List those departments that have no employees.

```
select e.FIRST_NAME, e.LAST_NAME, d.DEPARTMENT_NAME
from HR.DEPARTMENTS d left outer join HR.EMPLOYEES e
on (d.DEPARTMENT_ID = e.DEPARTMENT_ID);
```

11. Select the employees' names and the departments they work in. List those departments that have no employees. List those employees who have no dept. id too!

```
select e.FIRST_NAME, e.LAST_NAME, d.DEPARTMENT_NAME
from HR.DEPARTMENTS d full outer join HR.EMPLOYEES e
on (d.DEPARTMENT_ID = e.DEPARTMENT_ID);
```

Exercises for Class 7

1. Which country has the lowest population?

Remark: This can be done using a 2 step select:

- 1. step: Find the minimum value of the populations.
- 2. step: Find the country name

```
select name
from OLYMPICS.O_COUNTRIES
where POPULATION = (select min(population)
from OLYMPICS.O_COUNTRIES);
```

2. Who were born in the same city as Cseh László?

--the subquery (inner query) is executed first

```
select birthplace
from OLYMPICS.o_athletes
```

where name = 'Cseh László';

--main query: the result of the subquery is used here.

```
select *
from OLYMPICS.o_athletes
where birthplace = (select birthplace
                    from OLYMPICS.o_athletes
                    where name = 'Cseh László');
```

3. Which country has the highest population?

2 steps:

- 1. step: find the highest population value
- 2. step: find the corresponding country name

```
select name
from OLYMPICS.o_countries
where population = (select max(population)
                   from OLYMPICS.o_countries);
```

4. Which countries have higher population than that of Hungary?

```
select name, population
from OLYMPICS.o_countries
where population > (select population
                   from OLYMPICS.o_countries
                   where name = 'Hungary');
```

5. What is the name of the country where the youngest athlete comes from?

- 1. step: select the max birthdate
- 2. step: find the corresponding country_id
- 3. step: find the corresponding country name

```
select name
from OLYMPICS.o_countries
where c_id = (select country_id
              from OLYMPICS.o_athletes
              where birthdate = (select max(birthdate)
                                from OLYMPICS.o_athletes));
```

```
select c.name
from OLYMPICS.o_athletes a, OLYMPICS.o_countries c
where a.country_id = c.c_id and a.birthdate = (select max(birthdate)
                                                from OLYMPICS.o_athletes);
```

6. Who is the youngest athlete?

```
select name
from OLYMPICS.O_ATHLETES
```

where BIRTHDATE = (select min(birthdate)
from OLYMPICS.O_ATHLETES);

7. Which countries have more population than the average?

select name
from OLYMPICS.O_COUNTRIES
where population > (select avg(population)
from OLYMPICS.O_COUNTRIES);

8. Select the youngest athlete's name and birthdate in Hungarian date format and the length of their birthplace.

select name,to_char(birthdate,'YYYY.MM.DD'),length(birthplace)
from OLYMPICS.O_ATHLETES
where birthdate=(select max(birthdate)
from OLYMPICS.O_ATHLETES);

9. Which country has the highest population?

select name, population
from OLYMPICS.O_COUNTRIES
where population = (select max(population)
from OLYMPICS.O_COUNTRIES);

10. Which countries have higher population than Hungary?

select name
from OLYMPICS.O_COUNTRIES
where population > (select population
from OLYMPICS.O_COUNTRIES
where name = 'Hungary');

11. What is the name of the country from where the earliest-born athlete comes.

select a.name, c.name country_name
from OLYMPICS.O_ATHLETES a, OLYMPICS.O_COUNTRIES c
where a.country_id = c.c_id
and BIRTHDATE = (select max(BIRTHDATE)
from OLYMPICS.O_ATHLETES);

12. Which country has the most gold medals?

select c.name, m.gold
from OLYMPICS.O_COUNTRIES c, OLYMPICS.O_MEDAL_TABLE m
where c.C_ID = m.COUNTRY_ID
and gold = (select max(gold)
from OLYMPICS.O_MEDAL_TABLE);

13. Which country has the most medals?

```

select c.name, m.gold+m.silver+m.BRONZE num_of_medals
from OLYMPICS.O_COUNTRIES c, OLYMPICS.O_MEDAL_TABLE m
where c.C_ID = m.COUNTRY_ID
and m.gold+m.silver+m.BRONZE = (select max(gold+silver+bronze)
                                from OLYMPICS.O_MEDAL_TABLE);

```

14. Who were born before Cseh László?

```

select name, birthdate
from OLYMPICS.O_ATHLETES
where birthdate < (select birthdate
                  from OLYMPICS.O_ATHLETES
                  where name = 'Cseh László');

```

15. How many athletes were born before Cseh László?

```

select count(*)
from OLYMPICS.o_athletes
WHERE BIRTHDATE < (select birthdate
                  from OLYMPICS.o_athletes
                  WHERE name = 'Cseh László');

```

```

select count(born_before_laszlo.name)
from(select name
from OLYMPICS.O_ATHLETES
where birthdate < (select birthdate
                  from OLYMPICS.O_ATHLETES
                  where name = 'Cseh László')) born_before_laszlo;

```

16. What is the birthdate of the youngest athlete from those athletes who were born before Cseh László?

```

select *
from(select birthdate
from OLYMPICS.O_ATHLETES
where birthdate < (select birthdate
                  from OLYMPICS.O_ATHLETES
                  where name = 'Cseh László')) born_before_laszlo;

```

```

select max(born_before_laszlo.birthdate)
from(select birthdate
from OLYMPICS.O_ATHLETES
where birthdate < (select birthdate
                  from OLYMPICS.O_ATHLETES
                  where name = 'Cseh László')) born_before_laszlo;

```

17. Who is youngest athlete from those athletes who were born before Cseh László?

```

select name, birthdate

```

```

from OLYMPICS.O_ATHLETES
where birthdate = (select max(born_before_laszlo.birthdate)
                  from(select birthdate
                        from OLYMPICS.O_ATHLETES
                        where birthdate < (select birthdate
                                          from OLYMPICS.O_ATHLETES
                                          where name = 'Cseh László')) born_before_laszlo);

```

18. Find those African countries that have higher population than every European country.

```

select name, population
from OLYMPICS.O_COUNTRIES
where population > all(select population
                      from OLYMPICS.O_COUNTRIES
                      where continent = 'Europe')
and continent = 'Africa';

```

```

select name, population
from OLYMPICS.O_COUNTRIES
where population > (select max(population)
                  from OLYMPICS.O_COUNTRIES
                  where continent = 'Europe')
and continent = 'Africa';

```

19. Find those African countries that have higher population than at least one European country.

```

select name, population
from OLYMPICS.O_COUNTRIES
where population > any(select population
                      from OLYMPICS.O_COUNTRIES
                      where continent = 'Europe')
and continent = 'Africa';

```

```

select name, population
from OLYMPICS.O_COUNTRIES
where population > (select min(population)
                  from OLYMPICS.O_COUNTRIES
                  where continent = 'Europe')
and continent = 'Africa';

```

20. Who is the youngest from those who came from the largest country? Do not consider the athletes whose birthdate value is null.

```

select name
from OLYMPICS.o_athletes
where birthdate = (select max(birthdate)
                  from OLYMPICS.o_athletes a, OLYMPICS.o_countries c

```

```
where a.country_id = c.c_id
and c.area = (select max(area)
              from OLYMPICS.o_countries)
and birthdate is not null);
```

Exercises for Class 8

Remark:

Set operators: union, union all, intersect, minus

- *In the 2 sets the columns have to match in number and in their data types.*
- *The result set will contain the column names of the first set.*

- 1. List the names and the capitals in one set. If a capital name and a country name are the same, then list them only once.**

```
select name
from OLYMPICS.O_COUNTRIES
union
select capital
from OLYMPICS.O_COUNTRIES;
```

- 2. List the names and the capitals in one set.**

```
select name
from OLYMPICS.O_COUNTRIES
union all
select capital
from OLYMPICS.O_COUNTRIES;
```

- 3. List those European countries that won at least 5 gold medals.**

```
select c_id
from OLYMPICS.O_COUNTRIES
where continent = 'Europe'
INTERSECT
select country_id
from OLYMPICS.O_MEDAL_TABLE
where gold >= 5;
With the names:
```

```
select name
from OLYMPICS.o_countries
where c_id in (select c_id
               from OLYMPICS.o_countries
               where continent = 'Europe'
               INTERSECT
               select country_id
               from OLYMPICS.o_medal_table
               where gold >= 5);
```

where gold >= 5);

4. List the first 5 athletes.

```
select *
from OLYMPICS.O_ATHLETES
where rownum <=5;
```

5. List the first 5 athletes' names in alphabetical order.

WRONG: this return the first 5 athletes and the order by sorts them.

```
select name
from OLYMPICS.O_ATHLETES
where rownum <=5
order by name;
```

CORRECT: first sort then rownum

```
select name
from (select name
      from OLYMPICS.O_ATHLETES
      order by name)
where rownum <=5;
```

6. List the 2nd, 3rd, 4th and 5th athletes.

```
select *
from (select name, rownum r
      from OLYMPICS.o_athletes)
where r between 2 and 5;
```

7. List the first 5 athletes' names in alphabetical order. List the athletes' countries too

```
select name, country_name
from(
  select a.name, c.name as country_name
  from OLYMPICS.O_ATHLETES a, OLYMPICS.O_COUNTRIES c
  where a.COUNTRY_ID = c.c_id
  order by a.name --order by 1)
where rownum<=5;
```

**8. List the first 5 oldest athletes, their results and the corresponding sportevents.
You can assume that these athletes took part only in 1 event.**

```
select *
from (select a.name, r.place, se.name event_name
      from OLYMPICS.o_athletes a, OLYMPICS.o_results r,
```

```

OLYMPICS.o_sport_events se
where a.a_id = r.athlete_id and r.event_id = se.e_id
order by a.birthdate)
where rownum < 6;

```

9. List the number of medals earned by Cseh László.

```

select count(*) num_of_medals
from OLYMPICS.O_ATHLETES a, OLYMPICS.O_RESULTS r
where a.a_id = r.athlete_id and a.name = 'Cseh László' and place <= 3;

```

10. List those athletes' IDs who earned less medals than Cseh László.

```

select athlete_id, count(*)
from OLYMPICS.O_RESULTS
where place <= 3
group by ATHLETE_ID
having count(*) < (select count(*) num_of_medals
from OLYMPICS.O_ATHLETES a, OLYMPICS.O_RESULTS r
where a.a_id = r.athlete_id and a.name = 'Cseh László' and place <= 3);

```

11. Which country does the earliest-born athlete come from?

```

select a.name, c.name
from OLYMPICS.O_ATHLETES a, OLYMPICS.O_COUNTRIES c
where a.COUNTRY_ID = c.c_id and
birthdate = (select max(birthdate)
from OLYMPICS.O_ATHLETES);

```

12. List the IDs of those countries that are not in the medal table.

```

select c_id
from OLYMPICS.O_COUNTRIES
minus
select country_id
from OLYMPICS.O_MEDAL_TABLE;

```

13. List those countries that are not in the medal table.

```

select name
from OLYMPICS.O_COUNTRIES
where C_ID in (select c_id
from OLYMPICS.O_COUNTRIES
minus
select country_id
from OLYMPICS.O_MEDAL_TABLE);

```

```
select name
from OLYMPICS.O_COUNTRIES
where C_ID not in (select country_id from OLYMPICS.O_MEDAL_TABLE);
```

```
select name
from OLYMPICS.O_COUNTRIES
where not exists (select country_id
                  from OLYMPICS.O_MEDAL_TABLE
                  where country_id = c_id);
```

14. In Africa and in Asia which country has the highest population?

```
select name, population
from OLYMPICS.O_COUNTRIES
where (population, continent) in (select max(population), continent
                                   from OLYMPICS.O_COUNTRIES
                                   group by continent
                                   having continent in ('Africa', 'Asia'));
```

```
select name, population
from OLYMPICS.O_COUNTRIES c
where exists (select 1
              from OLYMPICS.O_COUNTRIES b
              group by continent
              having continent in ('Africa', 'Asia')
              and c.continent=b.continent
              and c.population=max(b.population));
```

```
select name
from OLYMPICS.O_COUNTRIES
where population = (select max(population)
                   from OLYMPICS.o_countries
                   where continent = 'Africa')
```

```
union
select name
from OLYMPICS.O_COUNTRIES
where population = (select max(population)
                   from OLYMPICS.o_countries
                   where continent = 'Asia');
```

Exercises for Class 9

SQL:

- DML (Data Manipulation Language): delete, insert, update, select
- DDL (Data Definition Language): create, drop, alter, truncate, rename
- Transaction Control Statements: commit, rollback, savepoint

1. Copy the content of the athletes table

```
create table athletes_copy  
as(select *  
from OLYMPICS.O_ATHLETES);
```

```
select *  
from athletes_copy  
order by A_ID;
```

2. Commit the previous statements

```
commit;
```

Remark: *The describe command provides a description of the specified table or view.*

```
describe athletes_copy;
```

3. Add new rows into athletes_copy:

```
insert into athletes_copy (a_id,name,birthdate,country_id,team,birthplace)  
values(800,'John Smith',sysdate,131,'no','Debrecen');
```

```
insert into athletes_copy (a_id,name,birthdate,country_id,team,birthplace)  
values(801,'Peter Smith',to_date('1990.02.02','YYYY.MM.DD'),131,'no','Debrecen');
```

Remark: *We do not need to specify the columns if we use all of them.*

```
insert into athletes_copy  
values(803,'Jake Smith',sysdate,131,'no','Budapest');
```

Remark: *birthdate will be NULL this way.*

```
insert into athletes_copy (a_id,name,country_id,team,birthplace)  
values(802,'Nora Smith',131,'no','Debrecen');
```

4. Delete John Smith from the athletes_copy table

```
delete from athletes_copy  
where name = 'John Smith';
```

5. Delete those individual athletes who came from Budapest.

```
delete from athletes_copy  
where team = 'no' and BIRTHPLACE = 'Budapest';
```

```
select *  
from athletes_copy  
order by A_ID;
```

6. Rollback the previous statements:

Remark: *It withdraws the previous statements and goes back to the previous commit.*

```
rollback;
```

7. Delete the earliest-born athlete

```
delete from athletes_copy  
where birthdate = (select max(birthdate)  
                  from athletes_copy );
```

8. Copy the Countries table and give it a new name.

```
create table countries_copy as  
select * from OLYMPICS.O_COUNTRIES;
```

9. Delete those players whose name starts with the letter 'D' or 'B' and come from Budapest.

```
delete from athletes_copy  
where (name like 'D%' or name like 'B%') and BIRTHPLACE = 'Budapest';
```

10. Delete those players who were born before 1992 and come from Budapest.

```
delete from athletes_copy  
where BIRTHPLACE = 'Budapest' and BIRTHDATE < to_date('1992','YYYY');
```

11. Delete the European players who were born before 1970.

```
delete from athletes_copy  
where a_id in (select a.a_id  
              from athletes_copy a, countries_copy c  
              where a.country_id = c.c_id and c.continent = 'Europe'  
              and a.birthdate < to_date('1970','YYYY'));
```

12. Delete those individual athletes who come from Budapest.

```
delete from athletes_copy  
where birthplace = 'Budapest' and team = 'no';
```

13. Delete the youngest athlete.

```
delete from athletes_copy  
where birthdate = (select max(birthdate)
```

from athletes_copy);

14. Copy the content of the countries table and give it a new name.

```
create table countires_copy
as (select * from OLYMPICS.o_countries);
```

15. Delete those athletes whose name starts with the letter 'D' or 'B' and come from Budapest.

```
delete from athletes_copy
where (name like 'D%' or name like 'B%') and birthplace = 'Budapest';
```

16. Delete the Hungarian teams.

```
delete from athletes_copy
where a_id in (select a_id
               from athletes_copy a, countries_copy c
               where a.country_id = c.c_id
               and team = 'yes' and c.name = 'Hungary');
```

```
delete from athletes_copy
where country_id in (select c_id
                    from countries_copy
                    where name = 'Hungary')
and team = 'yes';
```

17. Update: Raise the population of each European countries by 10%

```
update countries_copy
set POPULATION = population * 1.1
where continent = 'Europe';
```

18. Decrease the area of the Asian countries by 15%

```
update countries_copy
set area = area * 0.85
where continent = 'Asia';
```

19. Give another gold medal to Hungary (increase its number of gold medals by 1)

```
create table medal_table_copy
as (select * from OLYMPICS.O_MEDAL_TABLE);
```

```
update medal_table_copy
set gold = gold + 1
where country_id = (select c_id
                   from countries_copy
                   where name = 'Hungary');
```

```
where name = 'Hungary'
);
```

20. Let the population of Germany be the same as the population of Italy.

```
update countries_copy
set population = (select population
                  from countries_copy
                  where name = 'Italy')
where name = 'Germany';
```

21. Let Michael Phelps be 2 months older than Cseh László.

```
update athletes_copy
set birthdate = (select add_months(birthdate,-2)
                 from athletes_copy
                 where name = 'Cseh László')
where name = 'Michael Phelps';
```

```
update athletes_copy
set birthdate =
add_months((select birthdate from athletes_copy where name = 'Cseh László'),-2)
where name = 'Michael Phelps';
```

22. Delete those athletes who come from a country which did not earn a gold medal.

```
delete from athletes_copy
where country_id in (select country_id
                    from medal_table_copy
                    where gold = 0);
```

```
delete
from athletes_copy
where country_id in (select c_id
                    from countries_copy c, medal_table_copy mt
                    where c.c_id = mt.country_id
                    and mt.gold = 0);
```

Exercises for Class 10

Remark:

Table names and column names must:

- *Begin with a letter*
- *Be 1-30 characters long*
- *Contain only A-Z, a-z, 0-9, _, \$, and #*
- *Not duplicate the name of another object owned by the same user*

- *Not be an Oracle server-reserved word*

You must have:

- *The CREATE TABLE privilege*
- *A storage area*

You specify:

- *The table name*
- *The column name, column data type, and column size*

You can use constraints to do the following:

- *Enforce rules on the data in a table whenever a row is inserted, updated, or deleted from that*
- *table. The constraint must be satisfied for the operation to succeed.*
- *Prevent the deletion of a table if there are dependencies from other tables.*
- *You can name a constraint, or the Oracle server generates a name by using the SYS_Cn format.*

The following constraints can be defined:

- *NOT NULL Specifies that the column cannot contain a null value*
- *UNIQUE Specifies a column or combination of columns whose values must be unique for all rows in the table*
- *PRIMARY KEY Uniquely identifies each row of the table*
- *FOREIGN KEY Establishes and enforces a referential integrity between the column and a column of the referenced table such that values in one table match values in another table.*
- *CHECK Specifies a condition that must be true*

1. Create a table called o_sponsors. The columns of the table are:

- *id (max. 5 digit number),*
- *name (char. sequence, max. 30 characters long),*
- *headquarter (char. sequence, max 30 long).*
- *Let the primary key be the id column. The name column must be filled.*
- *The name and headquarter columns must be unique.*

```
create table o_sponsors(
id number(5) constraint o_s_pk primary key,
name varchar2(30) not null,
headquarter varchar2(30),
constraint o_uq unique (name, headquarter));
```

```
create table o_sponsors(
id number(5) primary key,
name varchar2(30) not null,
headquarter varchar2(30),
constraint sp_uq unique (name,headquarter));
```

```
create table o_sponsors(
id number(5),
```

```
name varchar2(30) not null,  
headquarter varchar2(30),  
constraint sp_pk primary key (id),  
constraint sp_uq unique (name,headquarter));
```

2. Set the length of the name in the o_sponsors table from 30 to 50.

Remark:

- *alter table: add new column or constraint, delete column or constraint*
- *modify the data type of a column or its length*

```
alter table o_sponsors  
modify (name varchar2(50));
```

3. Add a new column to the o_sponsors table. The name of the new column is date_of_founding and its data type is date.

```
alter table o_sponsors  
add date_of_founding date;
```

4. Delete the date_of_founding column from the sponsors table.

```
alter table o_sponsors  
drop column date_of_founding;
```

5. Check the result of the alter table command.

```
describe o_sponsors;
```

6. Insert 2 rows into the table:

```
insert into o_sponsors (id, name, headquarter)  
values (1,'MOL Nyrt.', 'Budapest');  
insert into o_sponsors (id, name, headquarter)  
values (2,'Magyar Suzuki Zrt.', 'Esztergom');  
Remark: violating a constraint (primary key)  
insert into o_sponsors (id, name, headquarter)  
values (2,'BMW', 'Debrecen');
```

7. Create a savepoint in the session called create_sponsors

```
savepoint create_sponsors;
```

8. Delete the content of the athletes_copy table

```
TRUNCATE table athletes_copy;
```

```
select *  
from athletes_copy;
```

9. Insert the athletes who come from Debrecen into the athletes_copy table

```
insert into athletes_copy(a_id,name,birthdate,country_id,team,birthplace)
select *
from OLYMPICS.O_ATHLETES
where birthplace = 'Debrecen';
```

10. Insert the Asian teams in the athletes_copy table.

```
insert into athletes_copy(select a.*
    from OLYMPICS.o_athletes a, OLYMPICS.o_countries c
    where a.country_id = c.c_id
    and a.team = 'yes' and c.continent = 'Asia');
```

11. Delete the Asian teams from the athletes_copy table.

```
delete from athletes_copy
where a_id in (select a_id
    from athletes_copy a, OLYMPICS.o_countries c
    where a.country_id = c.c_id
    and a.team = 'yes' and c.continent = 'Asia');
```

12. Set the birthdate of each athlete to the sysdate.

```
update athletes_copy
set birthdate = sysdate;
```

13. Set the birthdate of each row in the athlete_copy table to the corresponding birthdate value stored in the OLYMPICS.O_ATHLETE table.

```
update athletes_copy ac
set BIRTHDATE = ( select birthdate
    from OLYMPICS.O_ATHLETES a
    where a.a_id = ac.a_id);
```

```
update athletes_copy ac
set ac.birthdate = (select birthdate
    from OLYMPICS.o_athletes
    where a_id = ac.a_id);
```

14. Set the population of each European country to the minimum value of the population of the Asian countries.

```
update COUNTRIES_COPY
set population = (SELECT min(population) FROM COUNTRIES_COPY where continent = 'Asia')
```

where CONTINENT = 'Europe';

select *
from COUNTRIES_COPY;

15. Create a table containing those Asian countries whose population is higher than the average population of the Asian countries.

```
create table asian_countires
as (select *
from countries_copy
where population > (select avg(population)
                    from countries_copy
                    where continent = 'Asia')
and continent = 'Asia');
```

select *
from asian_countires;

16. Delete the previously created table

drop table asian_countires;

17. List those individual athletes (from the OLYMPICS.O_ATHLETES table) who are older than the youngest Hungarian athlete.

```
select *
from OLYMPICS.O_ATHLETES
where team = 'no'
and birthdate < (select max(birthdate)
                from OLYMPICS.O_ATHLETES a, OLYMPICS.O_COUNTRIES c
                where a.country_id = c.c_id
                and c.name = 'Hungary');
```

18. rollback to the previously defined savepoint

rollback to create_sponsors;

19. Create a table called o_sponsorships. The columns are:

- athlete_id
- event_id
- sponsor_id, which is a foreign key referencing to the primary key of the o_sponsors table.
- sum (max. 10 digit number, its default value is 1.000.000 and it cannot be less than 50000)
- s_date (date type, must be filled).
- The athlete_id and the event_id combine a foreign key referencing to the primary key of the o_results table

- The primary key of the table contains the following columns:
- athlete_id, event_id, sponsor_id, s_date.

```
create table o_sponsorships
(athlete_id number(5),
 event_id number(5),
 sponsor_id number(5) constraint o_sp_fk1 references o_sponsors (id),
 sum number(10) default 1000000,
 s_date date not null,
 constraint o_sp_pk primary key (athlete_id,event_id,sponsor_id,s_date),
 constraint o_sp_fk2 foreign key (athlete_id, event_id)
 references O_RESULTS (athlete_id, event_id),
 constraint o_sp_ch check (sum>=50000));
```

20. Copy the OLYMPICS.O_RESULTS table.

```
create table o_results
as (select * from OLYMPICS.O_RESULTS);
```

21. Let the primary key in the results table the athlete_id and the event_id together.

```
alter table o_results
add constraint o_r_pk primary key (athlete_id, event_id);
```

```
select *
from o_results;
```

22. Drop the o_sponsors table

```
drop table o_sponsors;
```

23. Create a synonym for the sport_events table.

Remark: Using synonyms, we can use the short name defined in the synonym.
create synonym o_e for OLYMPICS.O_SPORT_EVENTS;

Public synonyms: Public synonyms are accessible to all users.
CREATE PUBLIC SYNONYM athletes for OLYMPICS.O_ATHLETES;

```
select *
from o_e;
```

24. Create a sequence of numbers starting from 1

```
create sequence f_seq;
```

25. Delete the previously defined sequence

```
drop sequence f_seq;
```

26. Select the next value of the sequence

```
select f_seq.nextval from dual;  
select f_seq.nextval from dual;
```

27. Select the current value of the sequence

```
select f_seq.currval from dual;
```

28. Insert rows into the o_sponsors table using the values of the previously defined sequence

```
insert into o_sponsors (id, name, headquarter)  
values (f_seq.nextval, 'UNIDEB', 'Debrecen');  
insert into o_sponsors (id, name, headquarter)  
values (f_seq.nextval, 'ELTE', 'Budapest');
```

```
select *  
from o_sponsors;
```

29. Create a sequence of numbers starting from 10 to 1000.

```
create sequence f_seq2  
minvalue 10  
maxvalue 1000  
INCREMENT by 2;
```

```
select f_seq2.nextval from dual;  
select f_seq2.nextval from dual;  
select f_seq2.currval from dual;
```

30. Create a view that contains the athletes coming from Debrecen

```
create view athletes_from_deb as  
select *  
from OLYMPICS.O_ATHLETES  
where birthplace = 'Debrecen';
```

```
select *  
from athletes_from_deb;
```

31. Selecting objects of a user

```
select *  
from user_tables;
```

```
select *  
from all_tables;
```

```
select *  
from user_synonyms;
```

```
select *  
from user_sequences;
```

```
select *  
from user_views;
```

```
select *  
from all_sequences;
```

```
select *  
from v$version;
```

Remark:

- *grant selecting permission to a user*
- *permissions: select, delete, update*

32. Give a select permission to the “User”

```
grant select on o_sponsors to “User”;
```

33. Revoke the given permission

```
revoke select on o_sponsors from “User”;
```

Exercises for Class 11

An Oracle VIEW, in essence, is a virtual table that does not physically exist. Rather, it is created by a query joining one or more tables.

1. create a view containing the European teams.

```
create view european_teams as  
(select a.*  
from OLYMPICS.o_athletes a, OLYMPICS.o_countries c  
where a.country_id = c.c_id and a.team = 'yes'  
and c.continent = 'Europe');
```

```
select *  
from european_teams;
```

2. delete the previously created view

```
drop view european_teams;
```

3. How many Asian teams are there? Create a view that contains this value.

```
create view num_of_asian_teams as
(select count(*) num_of_asian_teams
from OLYMPICS.o_athletes a, OLYMPICS.o_countries c
where a.country_id = c.c_id and a.team = 'yes' and c.continent = 'Asia');

select *
from num_of_asian_teams;
```

4. delete the previously created view

```
drop view num_of_asian_teams;
```

5. List the country for which the number of cities is the highest.

```
select country_name, count(*)
from HR.countries c, HR.locations l
where c.country_id = l.country_id
group by country_name
having count(*) = (select max(count(*))
                    from HR.countries c, HR.locations l
                    where c.country_id = l.country_id
                    group by country_name);
```

6. List the country for which the number of cities is 4.

```
select country_name, count(*)
from HR.countries c, HR.locations l
where c.country_id = l.country_id
group by country_name
having count(*) = 4;
```

7. In which country does that employee work who earns the most?

```
select first_name || ' ' || last_name as FULL_NAME
from HR.countries c, HR.locations l, HR.departments d , HR.employees e
where e.department_id = d.department_id and d.location_id = l.location_id and
l.country_id = c.country_id and salary = (select max(salary)
                                           from HR.employees);
```

create a copy of the HR.EMPLOYEES table

```
create table employees_copy as
(select * from HR.employees);
```

8. Let the employee_id be the primary key

```
alter table employees_copy  
add primary key (employee_id);
```

```
alter table employees_copy  
add constraint ec_pk primary key (employee_id);
```

9. Create a table called Student

The columns of this table are:

- ID: number max 5 digits
- Name: char. sequence max 50, must be filled
- B_Date: date, must be filled
- Parent_ID: number max 6 digits
- Height: number max 3 digits.

The primary key of this table is the ID column.

The height values cannot be more than 250. The default value is 170 for the height.

The Parent_ID is a foreign key referencing to the primary key of the HR.EMPLOYEES table.

```
create table students(  
id number(5),  
name varchar2(50) not null,  
b_date date not null,  
parent_id number(6),  
height number(3) default 170,  
constraint st_pk primary key (id),  
constraint st_fk foreign key (parent_id) references employees_copy(employee_id),  
constraint st_ch check(height <= 250));
```

Other way:

```
create table students(  
id number(5) primary key,  
name varchar2(50) not null,  
b_date date not null,  
parent_id number(6),  
height number(3) default 170,  
constraint st_fk foreign key (parent_id) references employees_copy(employee_id),  
constraint st_ch check(height <= 250));
```

```
select *  
from user_constraints  
where table_name = 'STUDENTS';
```

```
create sequence students_seq  
minvalue 100  
increment by 100;
```

```

insert into students
values(students_seq.nextval,'John',sysdate,108,180);
insert into students
values(students_seq.nextval,'Jake',sysdate-60,110,190);
insert into students(id,name,b_date,parent_id)
values(students_seq.nextval,'George',add_months(sysdate,-12),102);
insert into students
values(students_seq.nextval,'Bello',add_months(sysdate,-12),104,260);

select *
from students;

```

10. Select the parents' names and their children's names

```

select name, first_name || ' ' || last_name parent_name
from students s, employees_copy e
where s.parent_id = e.employee_id;

```

11. Delete those students, whose parent is Nancy Greenberg.

```

delete from students
where parent_id = (select employee_id
                   from employees_copy
                   where first_name = 'Nancy' and last_name = 'Greenberg');

```

12. Increase Jake's height by 5%.

```

update students
set height = height * 1.05
where name = 'Jake';

```

13. Let George's height be the same as Jake's

```

update students
set height = (select height from students where name = 'Jake')
where name = 'George';

```

14. List the country names and the corresponding number of medals (gold, silver, bronze separately) List those countries that did not earn any medals as well.

```

select name, gold, silver, bronze
from OLYMPICS.o_countries c left outer join OLYMPICS.o_medal_table mt
on (c.c_id = mt.country_id);

```

15. List the country names and the corresponding number of medals (together in one column) List those countries that did not earn any medals as well. If a country did not earn any medals, display 0.

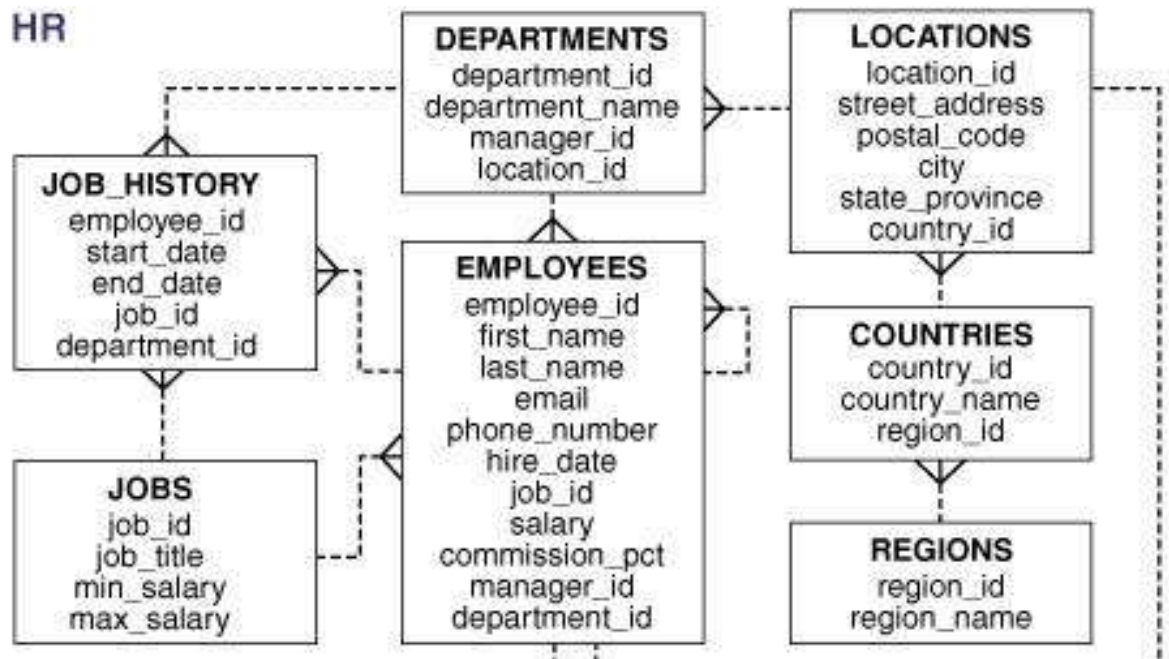
```

select name, nvl(gold+silver+bronze,0) num_of_medals
from OLYMPICS.o_countries c left outer join OLYMPICS.o_medal_table mt
on (c.c_id = mt.country_id);

```

Schemas

HR Schema:



OLYMPICS Schema:

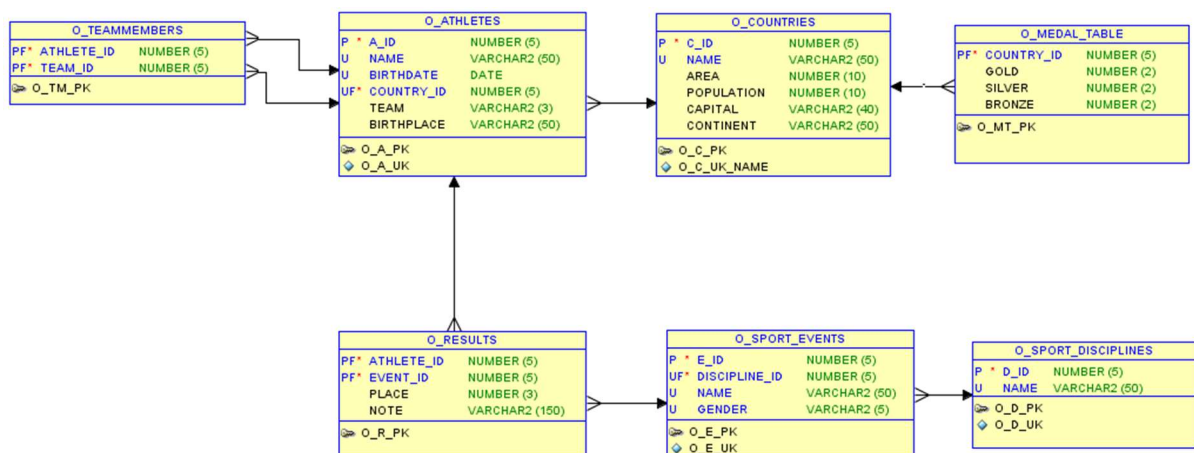


Table of Contents

Exercises for Class 1: select	1
Exercises for Class 2: where, comparison operators, logical operators, in, like, NULL, between.	2
Exercises for Class 3: numerical functions, character functions, date functions, nvl	5
Exercises for Class 4: aggregate functions, group by, having,	8
Exercises for Class 5: joins	11
Exercises for Class 6: outer joins, practice	14
Exercises for Class 7: subqueries	16
Exercises for Class 8: set operators, rownum	21
Exercises for Class 9: commit, rollback, create table, delete.....	25
Exercises for Class 10: create table, insert, views, sequences, grant, revoke,.....	28
Exercises for Class 11: practice.....	28
Schemas	39