University of Helsinki, Department of Computer Science,
Introduction to Databases, exam 7.3.2001

1.Let's consider the relations A(a,b,c) (cardinality n>0 tuples), B(a→A, d,e) (cardinality m>0 tuples) and C(a→A,d,e) (cardinality p>0 tuples). Notations: ∩ intersect, π projection, × Cartesian product, |×| join, σ selection, * natural join and |×[right outer join. An arrow indicates a foreign key and the relation it refers to.

a)Is it possible that the cardinality of $\pi_a(A)$ is less than the cardinality of $\pi_a(B)$?
No because a in B is a foreign key

b)Is it possible that the cardinality of B ∩ C is less than the cardinality of B * C ?
No, the results are equal

c)What is the cardinality of A $|\times|_{A.a=B.a}$ B?
m tuples (the number of tuples in B)

d)Is it possible that the cardinality of A $|\times[_{A.a=C.a}$ C differs from the cardinality of C$|\times[_{C.a=.A.a}$ A?
YES

e)Is B – C empty, if $\pi_a(B) - \pi_a(C)$ is empty?
NO

*f)*Is it true, that if the cardinality of $\pi_b(A)$ is *x* and the cardinality of $\pi_c(A)$ is *y* and *x>y*, then the cardinality of $\pi_{a,b}(A)$ is bigger than the cardinality of $\pi_{a,c}(A)$? *(2 points each)*
NO

Let's considet the following tables

```
chain(chain_id, chain_name)
cabin(cabin_id, region, size, address, chain→chain)
facilities(cabin_id, facility)
hobby_potential(cabin_id→ cabin, hobby_name, range_km)
customer(customer_id, name, address, phone)
reservation(cabin_id→cabin, year, week_number, customer_id→customer,
   date_reserved)
prices(cabin_id→cabin, year, season, price_for_week)
```

(season= winter, spring, summer, autumn)

2. Express the following queries in SQL. Specify a proper order for the results.

a)Make a list of cabins in Heinola region the weekly price of which is less than 1200 FIM in summer 2001.

```
select price_for_week, size, cabin_id, chain, address
from cabin, prices
where cabin.cabin_id=prices.cabin_id and
   cabin.region='Heinola' and
   prices.year=2001 and
   prices.season='summer' and
   prices.price_for_week <!200
   order by price_for_week
```

b)Make a list of unoccupied cabins in Lammi region on week 27 of year 2001.

```
select price_for_week, size, cabin_id, chain, address
from cabin, prices
where cabin.region ='Lammi' and
  cabin.cabin_id in
    (select cabin_id from prices
     where year=2001 and season='summer') and
  cabin.cabin_id not in
    (select cabin_id from reservation
     where year=2001 and
     week_number=27)
order by price_for_week;
```

c)Make a list of cabins in Heinola region that provide both a sauna and a barbecue and the potential for swimming within the range of 100 meters.

```
select price_for_week, size, cabin_id, chain, address
from cabin, facilities sauna, facilities grill, hobby_potential h
where cabin.region ='Heinola' and
   cabin.cabin_id =sauna.cabin_id and
   cabin.cabin_id =grill.cabin_id and
   cabin.cabin_id =h.cabin_id and
   sauna.facility='sauna' and
   grilli.facility='grill' and
   h.hobby_name='swimming' and
   h.range_km<0.1
order by price_for_week;
```

d)Produce the statistics of the amount of cabins in each region.

```
select region, count(*)
from cabin
group by region
order by region;

You may as well order the rows as
order by count(*) desc;
```

e)Which region has the most of cabins?

```
select region, count(*)
from cabin
group by region
having count(*) >=
  (select count(*) from cabin
   group by region);
```

*f)*Summer season contains the weeks 22 to 33. Produce a list of cabins that are occupied for more than 80% of the summer season. *(4 points for each)*

```
select price_for_week, size, cabin.cabin_id, count(*)/12
from cabin, reservation
where cabin.cabin_id=reservation.cabin_id and
   reservation.year=2001 and
   reservation.week_number between 22 and 33
group by price_for_week,size,cabin.cabin_id
having count(*)/12 > 0.8
order by price_for_week, size desc;
```

3. The chain 'Star' has decided to equip with a fridge all their cabins that have electricity but no fridge. This is to take place before the beginning of the summer season 2001. The weekly price for the improved cabins will be increased by 50 FIM starting from the beginning of the summer season 2001. Use SQL to make the required changes in the database *(6 points)*

```
update prices
set price_for_week =price_for_week+50
where
  (year=2001 and season in('summer','autumn','winter') or year>2001) and
   cabin_id in
      (select cabin_id from cabin
```

```
       where chain='Star' and
       cabin_id in
          (select cabin_id from facilities where facility='electricity')
       and
       cabin_id not in
          (select cabin_id from facilities where facility='fridge');
insert into facilities
  select cabin_id,'fridge'
  from cabin
  where chain='Star' and
     cabin_id in
         (select cabin_id from facilities where facility='electricity')
     and
     cabin_id not in
         (select cabin_id from facilities where facility='fridge')
;
commit;
```

4. Explain the concept database transaction. Include examples in your answer. *(8 points)*

Essentials:

- **a collection of operation that together for a single unit**
- **is executed completely or not at all**
- **outsiders never se intermediate results**
- **commit starts and ends**
- **results become public after commit they are also permanently stored**
- **may be cancelled prior to commit with rollback**

5. ***NOTE:*** *This task should be answered* ***only by the attendants of the final exam and by the students who want to substitute their practice points*** *with the point of this task (28 practice tasks, done in schedule, give the maximum of 10 points, and below that each point corresponds to about 2,5 tasks (rounded)).*

Let's consider the relation

`Student(student_id, name, major_id, major_name, credits_achieved)`

What is the meaning of the functional dependencies

`major_id`→`major_name` and `major_name`→`major_id`?

In this relation `student_id` determines functionally all the attributes of the relation. Is the relation in Boyce-Codd normal form? Justify your answer. *(10 points)*

Major_id uniquely determines major_name, i.e. when we know the id tor the major we may get only one related name. Names are also unique, i.e the same name is attached to only one id.

This relation is not in Boyce-Codd normal form. The key of the relation is sudent_id. This means that student_id -> name, major_id, major_name and credits_achieved. B-C presupposes that there are no dependencies the determinant of which does not contain the key. For example major_id→major_name violates this.