Database (Individual study point exercises - 14 study points)

Hand-in: In moodleroom. Solution document must contain headings for each exercise solution with solutions under each heading. Hand in must be in pdf format.

Deadline: Sunday February 19th at 23.59

Exercise 1

a) Underline all primary keys and use **bold** to indicate foreign keys, for these tables:

employees (eno, ename, zip, hdate)
parts (pno, pname, qoh, price, olevel)
customers (cno, cname, street, zip, phone)
orders (ono, cno, eno, received, shipped)
odetails (ono, pno, qty)
zipcodes (zip, city)

Explanations of attribute names:

hdate in employees table:	hire date
qoh in parts table:	quantity on hand (dk: antal på lager)
olevel in table parts:	order level (e.g. min. amount in stock before reordering)

b) Write an SQL-script with all necessary statements to create tables in exercise 1.a

- Define appropriate data types
- Define primary keys and foreign keys
- Assign necessary integrity constraints (not null, default)
- Run the script to create the database

---

c) Run the script "insert_mail.sql" (can be found on github), which populates the database. Hint: If the script fails, there might be a mismatch in the data types or constraints between the database that you just created and the insert statements in the SQL script. Adjust the script or the database tables.

Exercise 2

The following SQL statements will generate errors. Identify the cause and fix the error. Write down what you did and why.

a) `insert into employees values (1002, 'Olsen', 67226, '2006-09-13');`
b) `insert into odetails values (1020, 10900);`
c) `insert into employees values (1004, 'Jensen', 66666, '2006-09-15');`
d) `insert into parts values (11000, Harry Potter, 12, 23.25, 12);`
e) `insert into parts values (11001, 'Marx Brothers', 10, -22.99, 20);`
f) `update zipcodes set values (city='Los Angeles') where zip=67226;`

Exercise 3

Insert 1 additional row in each table. Run the 6 statements in a sequence, so that the integrity of the database is maintained at all time.

Exercise 4

Write queries to retrieve the results requested below from the database:

1. The names of all customers
2. The names of products of which there are at least 150 pieces in stock
3. Names and zip codes of all customers with a phone number which ends with '55'.
4. Names of products which cost less than 18.00
5. The name and city of all customers
6. Order numbers for orders made by an employee named 'Jones'
7. Customer name and order number for all orders, where the customers address begins with '1'.
8. All information about employees and the cities they live in. Include information about cities without employees (Hint: outer join).
9. Customer names and order numbers for all orders with customer that live in "Los Angeles"
10. A list with name, quantity, price and total price for all products on the order with order number 1020.
11. The price of all orders combined.
12. Order numbers for orders which have not yet been delivered.
13. The order number, number of line items, the customer's name, street, zip code and city

EXTRA (not mandatory)

14. The names of customers who have made an order with an employee who lives in 'Los Angeles'
15. Order number and total price of the entire order – for all orders

---

Exercise 5 (Only mandatory for red students)

Views are virtual tables defined as queries and operate on data from one or more tables. You can read about it at: http://www.mysqltutorial.org/mysql-views-tutorial.aspx, in particular the first three sections (Intro, Views in MySQL, Creating Views in MySQL)

a) Create a view from the employee table, but only with rows that have a zip code greater than 60000. Test the view with a "SELECT * FROM … "

b) Create another view also from the employee table, but only with the columns employee number and employee name. Test the view with a "SELECT * FROM … "

c) Create a view which includes employee numbers and names and the number of orders each employee has created. Test the view with a "SELECT * FROM … "

Example of View creation:

```
CREATE VIEW StaffBranch001
   AS
SELECT staffNo, name, position
   FROM Staff
   WHERE branchNo = 'B001';
```

Exercise 6 (Only mandatory for red students)

a) Consider the use of the unique constraint on non-primary key columns to prevent duplicate values in one or more columns.

b) Consider the use of ON DELETE CASCADE referential action for a foreign key to delete data from a child table automatically when you delete data from a parent table.