Exercise Sheet 2 (WS 2020) – Sample Solution

3.0 VU Datenmodellierung / 6.0 VU Datenbanksysteme

About the exercises

General information

In this part of the exercises you are asked to solve tasks in the areas of SQL, functional dependencies, and relational normal forms.

We recommend you to solve the problems **on your own** (it is a good preparation for the exam – and also for your possible future job – to carry out the tasks autonomously). Please note that if we detect duplicate submissions or any plagiarism, both the "original" and the "copy" will be awarded 0 points.

Your submission must consist of a single, typeset PDF document (max. 5MB). **We do not accept PDF files with handwritten solutions.**

In total there are 7 tasks and at most 15 points that can be achieved on the entire sheet.

Deadlines

until 11.11.	**12:00 pm**:	Solve the SQL tasks with the online-tool
until 18.11.	**12:00**:	Upload your solution to TUWEL
from 30.11.	13:00:	Evaluation and feedback is available in TUWEL

SQL Exercise

Note that the **mandatory** SQL exercise takes place in parallel to this exercise sheet. The SQL exercise takes place in its own online tool and is not part of the exercise sheet.

You can access the online tool in TUWEL: Select **eSQL Tool** in the section "SQL exercise". No additional password is needed, you are authenticated via TUWEL. *Please note:* Unfortunately, the interface of the online tool is currently available only in German. However, task descriptions and the description of the database are available in English.

Caution!
Separate deadline for finishing the SQL tasks:
Wednesday, November 11, 2019, 12:00 pm!

Further questions – TUWEL forum

If you have any further questions concerning the contents or organization, do not hestitate to ask them on TUWEL forum. **Under no circumstances should you post (partial) solutions on the forum!**

Changes due to COVID-19

Due to the ongoing situation with COVID-19 we will not offer in-person office hours for the exercise sheets. If you have technical issues, trouble understanding the tasks on this sheet, or other questions please use the TUWEL forum.

We also recommend that you get involved in the forum and actively discuss with your colleagues on the forum. From experience we believe that this helps all parties in the discussion greatly to improve their understanding of the material.

Normal Forms

Task 1 (Functional Dependencies) [1 point]

Consider the following relational schema

`Boat (Type, Length, Rooms, Sail, Name, Hull)`

with the following instance (sorted by `Type`):

Boat					
Type	**Length**	**Rooms**	**Sail**	**Name**	**Hull**
Barge	8	1	yes	Atafu	1
Canoe	2	0	no	Lihou	1
Canoe	3	0	no	Truk	1
Dinghy	6	0	yes	Maro	1
Houseboat	48	3	no	Ari	1
Katamaran	36	2	yes	Koror	2
Katamaran	42	5	yes	Ladi	2
Pinnace	18	1	yes	Atafu	1
Pinnace	19	1	yes	Masabu	1
Yacht	30	3	no	Deahu	1
Yacht	34	3	no	Maro	1
Yacht	40	5	no	Palau	2

Check for each functional dependency below whether it is satisfied by the given instance or not. For each FD, provide an answer (yes/no). If a FD is not satisfied, provide also a counter example. If a FD is satisfied, provide a tuple that, by adding it to the instance, would lead to a violation of the FD.

(a) `Type` $\rightarrow$ `Hull`. **No**

The last two tuples in the table share the `Type` *Yacht*, but have different values for their `Hull` attributes.

(b) `Length` $\rightarrow$ `Type` **Yes**

Every tuple with an an existing length but a different type would work. For example a tuple $(\text{Canoe}, 8, \dots)$, which would break the FD in combination with the tuple $(\text{Barge}, 8, \dots)$.

(c) `Name, Rooms` $\rightarrow$ `Sail`.

Yes

Only two names are not unique: *Atafu* and *Maro*. For all other tuples the FD is trivially satisfied. In the *Atafu* case we see that the pair (Atafu, 1) for attributes `Name` and `Rooms`, always has value *yes* for `Sail`.

We can break the FD, for example, with the tuple

$$(\star, \star, 5, yes, Palau, \star)$$

where $\star$ is a placeholder for the unimportant values (you can use arbitrary values here and the FD would still break). Together with the tuple in the last row of the table this new tuple is inconsistent with the FD for pair (Palau, 5).

(d) `Rooms, Sail` $\rightarrow$ `Name`. **No**

The FD is broken many times, for example by both tuples of `Type` *Canoe*.The relevant part for why the FD is unsatisfied there is highlighted below:

$$(\star, \star, 0, nein, \textbf{Lihou}, \star)$$
$$(\star, \star, 0, nein, \textbf{Truk}, \star)$$

(e) `Length, Hull` $\rightarrow$ `Rooms`.

Yes

Note that all values for `Length` are unique. Thus, all pairs of `Length`, `Hull` values in the database are also unique. From this we see that the FD is definitely satisfied.

To breka the FD, the `Length` values can not be unique. We again construct a unsatisfying case by creating a conflict with the last row of the table with the following new tuple.

$$(\star, 40, 1, \star, \star, 2)$$

(f) `Name, Type` $\rightarrow$ `Rooms, Hull`. **Yes**

Again we create a unsatisfying database by creating a conflict with the last row. Note that more attributes on the right side of the FD mean that there are more different ways to unsatisfy the FD. For example, adding tuple

$$(Yacht, \star, 1, \star, \text{Palau}, 2)$$

or

$$(Yacht, \star, 5, \star, \text{Palau}, 1)$$

both make the FD unsatisfied

Task 2 (Equivalence of Functional Dependencies)	[2 points]

(a) Consider the following relational schema $UVWXYZ$ and two sets F_1 and F_2 of functional dependencies.

$$F_1 = \{V \rightarrow WY, UX \rightarrow UV, Y \rightarrow UXV, XV \rightarrow Y, UX \rightarrow Z\}$$
$$F_2 = \{V \rightarrow WY, UX \rightarrow UV, Y \rightarrow UX, XVW \rightarrow Y, Y \rightarrow WZ\}$$

Are F_1 and F_2 equivalent? Please explain your answer using the closures of F_1 and F_2 and show your reasoning.

Lösung:
WWe show the equivalence by arguing $F_1 \subseteq F_2^+$ and $F_1^+ \supseteq F_2$. In both cases we argue, for every FD, that it is part of the hull of the respective other set of FDs. (We only mention the non-trivial inclusions below)

For $F_1 \subseteq F_2^+$

$Y \to UXV$ Because of $Y \to UX$ we have $UX \subseteq \text{AttrH}(Y, F_2)$. From $UX \to UV$ we can then deduce $UVX \subseteq \text{AttrH}(Y, F_2)$, hence $F_2 \models X \to UXV$.

$XV \to Y$ From $V \to WY$ we see $VWXY \subseteq \text{AttrH}(XV, F_2)$. Then, consider $VWX \to Y$ to derive $Y \subseteq \text{AttrH}(XV, F_2)$. We see $F_2 \models XV \to Y$.

$UX \to Z$ As above, we derive $Z \subseteq \text{AttrH}(UX, F_2)$ by expanding the known hull, step-by-step wit the following FDs: First $UX \to UV$, then $V \to WY$, and finally $Y \to WZ$.

For $F_2 \subseteq F_1^+$

$Y \to UX$ Immediate due to $Y \to UXV \in F_1$.

$XVW \to Y$ Immediate due to $V \to WY \in F_1$.

$Y \to WZ$ First $Y \to UXV$ to see $UXV \in \text{AttrH}(Y, F_1)$. Then, via $UX \to Z$ we also get $Z \in \text{AttrH}(Y, F_1)$, and finally because $V \to WY$ we have $W \in \text{AttrH}(Y, F_1)$. Hence, $F_1 \models Y \to WZ$.

(b) Consider the set F_2 of functional dependencies from task a). Please show using the Armstrong axioms that $F_2 \models \{UX \to VZ\}$ holds (show your reasoning).

Lösung:
To show that $F_2 \models \{UX \to VZ\}$ we have to derive FD $UX \to VZ$ from F_2 using the Armstrong-axioms F_2. One such derivation is given below.

Via decomposition we can, in a preliminary step, derive $UX \to V$ from $UX \to UV$. We use this to shorten two branches in the proof below.

$$\dfrac{\dfrac{\dfrac{\dfrac{\overset{[\text{See above}]}{UX \to V} \quad \dfrac{\overset{[\text{Given}]}{V \to WY}}{V \to Y}\,\text{Decomposition}}{UX \to Y}\,\text{Transitivity} \quad \overset{[\text{Given}]}{Y \to WZ}}{UX \to WZ}\,\text{Transitivity} \quad \overset{[\text{See above}]}{UX \to V}}{UX \to VWZ}\,\text{Union}}{UX \to VZ}\,\text{Decomposition}$$

Task 3 (Minimal Cover)	[2 points]

Provide a canonical cover of the sets $\mathcal{F}_1, \mathcal{F}_2$ of functional dependencies over the relational schema $\mathcal{R} = ABCDEFG$ and document your reasoning.

(a) $\mathcal{F}_1 = \{A \to DEG, BE \to F, CG \to ECF, E \to CA, E \to AE, FG \to AB, G \to C\}$

(b) $\mathcal{F}_2 = \{AB \to C, EFG \to CDE, CD \to F, A \to F, FGA \to C, E \to A\}$

Lösung:

(a) A canonical cover can be computed using the following four steps (the solution is not unique)

- **decomposition:**

  with decomposition we obtain
  $\{A \to D, A \to E, A \to G, BE \to F, CG \to E, CG \to C, CG \to F, E \to C, E \to A, E \to A, E \to E, FG \to A, FG \to B, G \to C\}$, which is equivalent to (as it is a set)
  $\{A \to D, A \to E, A \to G, BE \to F, CG \to E, CG \to C, CG \to F, E \to C, E \to A, E \to E, FG \to A, FG \to B, G \to C\}$

- **left reduction:**

  We have to check for each FD whether an attribute on the left side is superfluous and therefore can be omitted such that the resulting set of FDs is equivalent to the original set of FDs.

  Already left reduced are $\{A \to D, A \to E, A \to G, E \to C, E \to A, E \to E, G \to C\}$ and therefore have not to be considered. We focus on the set $\{BE \to F, CG \to E, CG \to C, CG \to F, FG \to A, FG \to B\}$.

  $BE \to F$ can be reduced to $E \to F$, as F is in the attribute closure of $(\mathcal{F}_1, E)$. $CG \to E$ can be reduced to $G \to E$, as E is in the attribute closure of $(\mathcal{F}_1, G)$ (attention: we have to consider the current - reduced - set here). $CG \to C$ can be reduced to the trivial FD $C \to C$, even without constructing the attribute closure. This FD is trivial and therefore can be cancelled here already. $CG \to F$ can be reduced to $G \to F$, as F is in the attribute closure of $(\mathcal{F}_1, G)$. $FG \to A$ can be reduced to $G \to A$, as A is in the attribute closure of $(\mathcal{F}_1, G)$. $FG \to B$ can be reduced to $G \to B$, as B is in the attribute closure of G.

  The set F_r of FDs after the left reduction is
  $\{A \to D, A \to E, A \to G, E \to C, E \to A, E \to E, G \to C, E \to F, G \to E, C \to C, G \to F, G \to A, G \to B\}$.

- right reduction:

  Here we have to check for every FD whether it can be deleted such that the resulting set of FDs is equivalent to the original one. Therefore we have to check whether the corresponding FD can be derived from the remaining FDs or not.

  For $A \to D$ this is not the case, as it is the only FD which has D on the right side. $A \to E$ can be removed, as E is in the attribute closure of (F_r', A), where $F_r' = F_r \setminus \{A \to E\}$. We obtain $F_r = F_r'$.

  $A \to G$ cannot be reduced, but $E \to C$ can as C is in the attribute closure of (F_r'', E), where $F_r'' = F_r \setminus \{E \to C\}$. We obtain $F_r = F_r''$.

  $E \to A$ and $G \to C$ cannot be reduced. $E \to E$ is a trivial FD and is therefore reduced. $E \to F$ can be reduced, because F is in the attribute closure of (F_r''', E), where $F_r''' = F_r \setminus \{E \to F\}$. We obtain $F_r = F_r'''$.

  $G \to E$ and $G \to F$ cannot be reduced. $C \to C$ is trivial and is thereofre reduced. $G \to A$ can also be reduced, as A is in the attribute closure of (F_r'''', G), where $F_r'''' = F_r \setminus \{G \to A\}$. We obtain $F_r = F_r''''$. $G \to B$ cannot be reduced.

The set of FDs after right reduction is

$$\{A \to D, A \to G, E \to A, G \to C, G \to E, G \to F, G \to B\}.$$

- **union:**

  The last step is to combine FDs with identical left sides with union. We obtain:

  $$\{A \to DG, E \to A, G \to BCEF.$$

(b) We consider the four steps to construct a canonical cover (the solution is not unique).

- **decomposition:**

  We obtain:
  $\{AB \to C, EFG \to C, EFG \to D, EFG \to E, CD \to F, A \to F, FGA \to C, E \to A\}$.

- **left reduction:**

  We have to check for each FD whether an attribute on the left side is superfluous and therefore can be omitted such that the resulting set of FDs is equivalent to the original set of FDs.

  This is not the case for $AB \to C$. $EFG \to C$ can be reduced to $EG \to C$, as C is in the attribute closure of EG. $EFG \to D$ can be reduced to $EG \to D$, as F is in the attribute closire of EG. $EFG \to E$ can be reduced to the trivial FD $E \to E$ without constructing the attribute closure. $CD \to F$ cannot be reduced, but $FGA \to C$ can be reduced to $GA \to C$.

  The set of FDs after left reduction is

  $$F_r = \{AB \to C, EG \to C, EG \to D, E \to E, CD \to F, A \to F, GA \to C, E \to A\}.$$

- right reduction:

  Here we have to check for every FD whether it can be deleted such that the resulting set of FDs is equivalent to the original one. Therefore we have to check whether the corresponding FD can be derived from the remaining FDs or not.

  $AB \to C$ cannot be reduced, but $EG \to C$ can as C is in the attribute closure of $(F_r \backslash (EG \to C), EG)$. $EG \to D$ cannot be reduced, but the FD $E \to E$ is a trivial FD and therefore can be deleted. All the remaining FDs cannot be reduced.

  Therefore, the set of FDs after right reduction is given by

  $$\{AB \to C, EG \to D, CD \to F, A \to F, GA \to C, E \to A\}.$$

- **union:**

  The last step is to combine FDs with identical left sides with union. We obtain:

  $$\{AB \to C, EG \to D, CD \to F, A \to F, GA \to C, E \to A\}.$$

Task 4 (Identifying Keys and Superkeys) [2 points]

For the following relational schemata with their functional dependencies, find *all keys* and *all superkeys*.

(a) $\mathcal{R} = ABCDE$
$F = \{AB \to DE, CD \to BE, E \to A, D \to C\}$

Lösung:
The keys are D and AB and BE. The set of super keys is

$$\begin{aligned}\{D, AB, BE, DA, DB, DC, DE,\\ ABC, ABD, ABE, BCE, BDE, ACD, ADE, BCD, CDE,\\ ABCD, ABDE, BCDE,\\ ABCDE\}\end{aligned}$$

(b) $\mathcal{R} = ABCDEFG$
$F = \{AB \to EF, CD \to G, F \to DG, E \to B\}$

Lösung:
The set of keys is

$$\{ABC, ACE\}.$$

The set of super keys is

$$\begin{aligned}\{ABC, ACE,\\ ABCD, ABCE, ABCF, ABCG, ACDE, ACEF, ACEG,\\ ABCDE, ABCDF, ABCDG, ABCEF, ABCEG, ABCFG, ACDEF, ACDEG, ACEFG\\ ABCDEF, ABCDEG, ABCDFG, ACDEFG\\ ABCDEFG\}\end{aligned}$$

Task 5 (Normal Forms) [2 points]

For each subtask, assume a relational schema $\mathcal{R}$ with its set $\mathcal{F}$ of functional dependencies. Please check, whether $\mathcal{R}$

- is in third normal form,
- in Boyce-Codd normal form,

and justify your answer.

Lösung:
Recall, for a FD of the form $\alpha \to B$, there are three properties that determine whether a relational schema R is in third normal form or Boyce-Codd normal form:

1. $B \in \alpha$, i.e., the FD is trivial;
2. α is a superkey of $\mathcal{R}$;
3. attribute B is contained in a key of $\mathcal{R}$.

A schema R is in Boyce-Codd normal form, if for every FD, property 1 or 2 is satisfied. The schema is in third normal form if every FD satsifies at least one of properties 1, 2, or 3.

(a) $\mathcal{R} = VWXYZ$,
$\mathcal{F} = \{XZ \to V, Z \to WY, VX \to WX, W \to YXZ\}$

Lösung:
We first determine the keys of $\mathcal{R}$. They are Z, VX, and W.

To decide the properties we first apply the decomposition rule to the FDs and then check the satisfied properties.

$$\mathcal{F} = \{\underbrace{AB \to C}_{2,3}, \underbrace{C \to A}_{2}, \underbrace{B \to D}_{2}, \underbrace{B \to B}_{1,2,3}, \underbrace{B \to A}_{2}, \underbrace{AC \to B}_{2,3}, \underbrace{AC \to D}_{2}\}.$$

Since every FD satisfies either proprety 1 or 2, we have that $\mathcal{R}$ is in BCND (and thus also in 3NF).

(b) $\mathcal{R} = UVWXYZ$
$F = \{UWZ \to UVY, XYZ \to W, VZ \to WXY, XY \to UZ,$
$UVW \to YW, UZ \to X\}$
The keys are given as VZ, UWZ, $UVWX$, UYZ, and XY.

Lösung:
We proceed the same as before by checking the properties for every decomposed FD.

$$\mathcal{F} = \{\underbrace{ACD \to B}_{2,3}, \underbrace{ACD \to C}_{2}, \underbrace{BDF \to C}_{2}, \underbrace{BDF \to E}_{2,3}, \underbrace{EF \to A}_{2,3}, \underbrace{EF \to B}_{2,3}, \underbrace{EF \to D}_{2,3}, \underbrace{ABE \to C}_{2}, \underbrace{ABE \to D}_{2,3}, \underbrace{ABC \to B}_{3}, \underbrace{ABC \to F}_{3}, \underbrace{AD \to A}_{1,2}, \underbrace{AD \to C}_{2}, \underbrace{ACF \to C}_{1}, \underbrace{ACF \to F}_{3}\}.$$

Since every FD satisfies at least one of the thee properties, but some satisfy **only** property (3), the schema is in 3NF but not in BCNF

Task 6 (Synthesis Algorithm) [3 points]

Consider the following relational schema and its functional dependencies:

$\mathcal{R} = ABCDEF$
$\mathcal{F} = \{BC \to A, A \to D, BE \to DF, EF \to C, E \to F, A \to B, B \to D\}$

We are looking for a lossless and dependency preserving decomposition in third normal form. Please apply the synthesis algorithm and show the results after every single step. Compute all keys of $\mathcal{R}$ and all relations of the decomposition.

Lösung:

1. Find the canonical cover:

$$\mathcal{F}_c = \{A \to B, B \to D, BC \to A, E \to CF\}$$

2. Determine all the candidate keys of $\mathcal{R}$ w.r.t. $\mathcal{F}_c$:

$$\{AE, BE\}.$$

3. Create a relation schema for every element of $\mathcal{F}_c$:

Schema	Satisfied FDs
$\mathcal{R}_1 = AB$	$\mathcal{F}_1 = \{A \to B\}$
$\mathcal{R}_2 = ABC$	$\mathcal{F}_2 = \{BC \to A, A \to B\}$
$\mathcal{R}_3 = BD$	$\mathcal{F}_3 = \{B \to D\}$
$\mathcal{R}_4 = CEF$	$\mathcal{F}_4 = \{E \to CF\}$

4. Eliminate schemas that are contained in other schemas.

   $\mathcal{R}_1$ is contained in $\mathcal{R}_2$ enthalten and can therefore be eliminated.

Schema	Satisfied FDs
$\mathcal{R}_2 = ABC$	$\mathcal{F}_2 = \{BC \to A, A \to B\}$
$\mathcal{R}_3 = BD$	$\mathcal{F}_3 = \{B \to D\}$
$\mathcal{R}_4 = CEF$	$\mathcal{F}_4 = \{E \to CF\}$

5. Test whether the candidate keys are part of a schema.

   None of them is part of a schema in our case. Thus, we create a new schema $\mathcal{R}_{\mathcal{K}}$, for candidate key AE:

Schema	Satisfied FDs
$\mathcal{R}_{\mathcal{K}} = AE$	$\mathcal{F}_{\mathcal{K}} = \emptyset$

6. Result:

Schema	Satisfied FDs
$\mathcal{R}_2 = ABC$	$\mathcal{F}_2 = \{BC \to A, A \to B\}$
$\mathcal{R}_3 = BD$	$\mathcal{F}_3 = \{B \to D\}$
$\mathcal{R}_4 = CEF$	$\mathcal{F}_4 = \{E \to CF\}$
$\mathcal{R}_{\mathcal{K}} = \underline{AE}$	$\mathcal{F}_{\mathcal{K}} = \emptyset$

Task 7 (Decomposition Algorithm) [3 points]

Consider the following relational schema with its functional dependencies and the list of all its keys:

$$\mathcal{R} = ABCDEF$$
$$\mathcal{F} = \{ABC \to B, AC \to DE, E \to C, F \to B\}$$

Keys: ACF, AEF

We are looking for a lossless decomposition into Boyce-Codd normal form. Please apply the decomposition algorithm and show the results after every single step. Compute all keys for all relations of the decomposition. Is the decomposition dependency preserving? If not, please provide the dependencies in $\mathcal{F}$ that got lost.

Hint: Compute for every decomposition the corresponding closures of FDs!

Lösung:

The three FDs

- $AC \to DE$,
- $E \to C$, and
- $F \to B$

violate the BCNF (they are not trivial and the left side contains no super key), the remaining FD satisfies the BCNF. There are three possible ways to decompose. We will illustrate one possibility here (the others are similar).

Remark: we illustrate in every hull $\mathcal{F}_i^+[\mathcal{R}_j]$ only the non trivial and left reduced FDs.

- We pick theFD $AC \to DE$ and obtain:

$$\mathcal{R}_1 = ACDE \qquad \mathcal{F}_1 = \mathcal{F}^+[\mathcal{R}_1] = \{AC \to DE, E \to C\}$$
keys: AC, AE
$$\mathcal{R}_2 = ABCF \qquad \mathcal{F}_2 = \mathcal{F}^+[\mathcal{R}_2] = \{ABC \to B, F \to B\}$$
keys: AFC

The schemata $\mathcal{R}_1$ and $\mathcal{R}_2$ are not in BCNF, as the FDs $E \to C$ and $F \to B$ violate the conditions of the BCNF. We decompose $\mathcal{R}_1$ after the only FD that violates the BCNF.

- We decompose after $E \to C$ to

$$\mathcal{R}_{1,1} = EC \qquad \mathcal{F}_{1,1} = \mathcal{F}_1^+[\mathcal{R}_{1,1}] = \{E \to C\} \qquad \text{keys: } E$$
$$\mathcal{R}_{1,2} = ADE \qquad \mathcal{F}_{1,2} = \mathcal{F}_1^+[\mathcal{R}_{1,2}] = \{AE \to D\} \qquad \text{keys: } AE$$

Now all relation schemata satisfy the BCNF.

We decompose $\mathcal{R}_2$ based on the only FD which violates the BCNF.

– We decompose after $F \to B$ to

$$\begin{array}{lll} \mathcal{R}_{2,1} = FB & \mathcal{F}_{2,1} = \mathcal{F}_2^+[\mathcal{R}_{2,1}] = \{F \to B\} & \text{keys: } F \\ \mathcal{R}_{2,2} = ACF & \mathcal{F}_{2,2} = \mathcal{F}_2^+[\mathcal{R}_{2,2}] = \{\} & \text{keys: } ACF \end{array}$$

Now all relation schemata satisfy the BCNF.

The decomposition is not dependency preserving, as the FG $AC \to DE$ is lost.