

Questão 01 - Parte 01

$A \rightarrow C$

$A \rightarrow D$

$E \rightarrow A$

$E \rightarrow H$

Show Steps



Step 1: Rewrite the FD into those with only one attribute on RHS. We obtain:

$A \rightarrow C$

$C A \rightarrow D$

$E \rightarrow A$

$E \rightarrow D$

$E \rightarrow H$

Step 2: Remove trivial FDs (those where the RHS is also in the LHS). We obtain:

Step 2: Remove trivial FDs (those where the RHS is also in the LHS). We obtain:

$A \rightarrow C$

$C A \rightarrow D$

$E \rightarrow A$

$E \rightarrow D$

$E \rightarrow H$

Step 3: Minimize LHS of each FD. We obtain:

$A \rightarrow C$

$A \rightarrow D$

$E \rightarrow A$

$E \rightarrow D$

$E \rightarrow H$

Step 4: Remove redundant FDs (those that are implied by others). We obtain:

Step 4: Remove redundant FDs (those that are implied by others). We obtain:

$A \rightarrow C$

$A \rightarrow D$

$E \rightarrow A$

$E \rightarrow H$

Questão 01 - parte2

Step 1: Rewrite the FD into those with only one attribute on RHS. We obtain:

$A \rightarrow C$

$A \rightarrow D$

$E \rightarrow A$

$E \rightarrow H$

Step 2: Remove trivial FDs (those where the RHS is also in the LHS). We obtain:

$A \rightarrow C$

$A \rightarrow D$

$E \rightarrow A$

$E \rightarrow H$

Step 3: Minimize LHS of each FD. We obtain:

$A \rightarrow C$

$A \rightarrow D$

$E \rightarrow A$

$E \rightarrow H$

Step 4: Remove redundant FDs (those that are implied by others). We obtain:

$A \rightarrow C$

$A \rightarrow D$

$E \rightarrow A$

$E \rightarrow H$

Questão 02 -

Show Steps



Step 1: Rewrite the FD into those with only one attribute on RHS. We obtain:

ssn → ename

ssn → bdate

ssn → address

ssn → dnumber

dnumber → dname

dnumber → dmgrssn

Step 2: Remove trivial FDs (those where the RHS is also in the LHS). We obtain:

ssn → ename

ssn → bdate

ssn → address

ssn → dnumber

dnumber → dname

dnumber → dmgrssn

Step 3: Minimize LHS of each FD. We obtain:

ssn → ename

ssn → bdate

ssn → address

ssn → dnumber

dnumber → dname

dnumber → dmgrssn

Step 4: Remove redundant FDs (those that are implied by others). We obtain:

ssn → ename

ssn → bdate

ssn → address

ssn → dnumber

dnumber → dname

dnumber → dmgrssn

Questão 3 -

3.1 -

Candidate Keys Found

- **A** **B**

Show Steps



Step 1: Find the minimal cover of FDs, which contains

A,B → C

A → D

A → E

B → F

F → G

F → H

D → I

D → J

Step 2: Find the set of attributes not on the RHS of any FD, which is NotOnRHS = {A,B}. Every CK must contain these attributes.

Step 3: NotOnRHS is a superkey, so it is the only candidate key

3.2 -

Check Normal Form



2NF
The table is not in 2NF.



3NF
The table is not in 3NF.



BCNF
The table is not in BCNF.

Show Steps

☐

3.3 -

Step 1: Find the minimal cover of FDs, which contains
A,B → C
A → D
A → E
B → F
F → G
F → H
D → I
D → J

Step 2: Find all candidate keys. The set of candidates keys is { (A,B), }.
The set of key attributes is: { A,B }.

Step 3: Merge FDs with same LHS and whose RHS are non-key attributes, we get the set F1 which contains:
A,B → C
A → E,D
B → F
F → H,G
D → J,I

Step 4: Check each FD in the set F1 for violation of 3NF, and split table accordingly.

Checking FD A,B → C
FD does not violate 3NF
Checking FD A → E,D
The FD violates 3NF as its LHS is not a superkey (and RHS is a set of non-key attributes).
The following 3NF table is obtained:

A,E,D
with FDs
A → D,E

Checking FD $B \rightarrow F$
The FD violates 3NF as its LHS is not a superkey (and RHS is a set of non-key attributes).
The following 3NF table is obtained:

B,F
with FDs
 $B \rightarrow F$

Checking FD $F \rightarrow H,G$
The FD violates 3NF as its LHS is not a superkey (and RHS is a set of non-key attributes).
The following 3NF table is obtained:

F,H,G
with FDs
 $F \rightarrow H,G$

Checking FD $D \rightarrow J,I$
The FD violates 3NF as its LHS is not a superkey (and RHS is a set of non-key attributes).
The following 3NF table is obtained:

D,J,I
with FDs
 $D \rightarrow J,I$

Step 5: Finally, add the following table into normalized 3NF table set (obtained by removing RHS attributes of FDs using which we produced a table):

A,B,C
with FDs

Step 5: Finally, add the following table into normalized 3NF table set (obtained by removing RHS attributes of FDs using which we produced a table):

A,B,C
with FDs
 $A,B \rightarrow C$

Questão 04 -

4.1

Candidate Keys Found

• **A** **B** **D**

Show Steps



Step 1: Find the minimal cover of FDs, which contains

$A,B \rightarrow C$
 $B,D \rightarrow E$
 $B \rightarrow F$
 $F \rightarrow G$
 $F \rightarrow H$
 $D \rightarrow I$
 $D \rightarrow J$

Step 2: Find the set of attributes not on the RHS of any FD, which is $\text{NotOnRHS} = \{A,B,D\}$. Every CK must contain these attributes.

Step 3: NotOnRHS is a superkey, so it is the only candidate key

4.2 -

Check Normal Form



2NF

The table is not in 2NF.



3NF

The table is not in 3NF.



BCNF

The table is not in BCNF.

Questão 5

Find Candidate Keys

candidate keys

Candidate Keys Found

- a d b
- a d c
- a d e

Questão 06 -

Find Candidate Keys

keys

Candidate Keys Found

- semester year courseno secno
- semester year dayshours roomno

Normalização 3NF

Attributes

courseeno courselevel credithours offeringdept

Functional Dependencies

courseeno → courselevel credithours offeringdept

Attributes

courseeno secno instructorssn semester year dayshours roomno noofstudents

Functional Dependencies

courseeno secno semester year → roomno dayshours

dayshours roomno semester year → secno courseeno instructorssn noofstudents

Questão 07-

Find Candidate Keys

Candidate Keys Found

- A B

candidate keys

Show Steps



Normalização 3NF

Attributes

A E D

Functional Dependencies

A → D E

able to 3NF
FDs attributes

B F

Functional Dependencies

B → F

Attributes

F H G

able to 3NF
g FDs

F H G

Functional Dependencies

F → H G

Attributes

D J I

Functional Dependencies

D → J I

Attributes

A B C

Attributes

D J I

Functional Dependencies

D → J I

Attributes

A B C

Functional Dependencies

A B → C

Questão 08-

Find Minimal Cover

A B → C

A → D

A → E

B → F

F → G

F → H

D → I

D → J

Normalização 3NF

Normalize to 3NF

Attributes

A B C

Functional Dependencies

A B → C

Convert this table to 3NF
preserving FDs

A E D

Functional Dependencies

A → D E

F → H G

Attributes

D J I

Functional Dependencies

D → J I

Attributes

A B

Functional Dependencies

Questão 08 -

Normalize to 3NF

Attributes

A B C

Functional Dependencies

A B → C

Normalize this table to 3NF
preserving FDs

A E D

Functional Dependencies

A → D E

Normalize this table to 3NF
preserving FDs

A E D

Functional Dependencies

A → D E

Attributes

B F

Functional Dependencies

B → F

Attributes

F H G

Functional Dependencies

B F

Functional Dependencies

B → F

Attributes

F H G

Functional Dependencies

F → H G

Attributes

D J I

Attributes

D J I

Functional Dependencies

D → J I

Attributes

(A B

Functional Dependencies

Questão 09 -

Normalize to BCNF

Attributes

(A B)

Functional Dependencies

Attributes

F G H

Functional Dependencies

F → G H

B F

Functional Dependencies

B → F

Attributes

A B C

Functional Dependencies

A B → C

Attributes

D I J

Functional Dependencies

Questão 10 -

☒ EDIT ATTRIBUTES

☐ LEARNING RESOURCES

☐ LOAD EXAMPLE

Functions

FIND A MINIMAL COVER

FIND ALL CANDIDATE KEYS

CHECK NORMAL FORM

NORMALIZE TO 2NF

NORMALIZE TO 3NF METHOD 1

NORMALIZE TO 3NF METHOD 2

NORMALIZE TO BCNF

About this tool

Normalize to BCNF

Attributes

a b c

Functional Dependencies

a b → c

Attributes

a b d

Functional Dependencies

Attributes

f g h

Functional Dependencies

f → g h

Attributes

b f

Functional Dependencies

b → f

Attributes

d i j

Functional Dependencies

d → i j

Attributes

b d e

Functional Dependencies

b d → e

Attributes

D I J

Functional Dependencies

D → I J

Attributes

A D E

Functional Dependencies

A → D E

Questão 11

Chaves

Find Candidate Keys

Candidate Keys Found

- A D B
- A D C
- A D E

Candidate keys

Normalização 3NF

Questão 12 -

Chaves Principais

Find Candidate Keys

Candidate Keys Found

- semester year courseno secno
- semester year dayshours roomno

Relação 3NF Método 2

Attributes

courseno offeringdept credithours courselevel

Functional Dependencies

courseno → offeringdept credithours courselevel

Attributes

courseno secno instructorssn semester year dayshours roomno noofstudents

Functional Dependencies

courseno secno semester year → roomno dayshours

dayshours roomno semester year → secno courseno instructorssn noofstudents

E relação BCNF

Attributes

courseno **offeringdept** **credithours** **courselevel**

Functional Dependencies

courseno \rightarrow **offeringdept** **credithours** **courselevel**

Attributes

courseno **secno** **instructorssn** **semester** **year** **dayshours** **roomno** **noofstudents**

Functional Dependencies

courseno **secno** **semester** **year** \rightarrow **roomno** **dayshours**

dayshours **roomno** **semester** **year** \rightarrow **secno** **courseno** **instructorssn** **noofstudents**

Normalize to 3NF

Attributes

A **B** **C** **D** **E**

Functional Dependencies

A **B** \rightarrow **C**

C **D** \rightarrow **E**

D **E** \rightarrow **B**

is table to 3NF
using FDs

Relações BNCF

A B C

Functional Dependencies

A B → C

Attributes

D E B

Functional Dependencies

D E → B

Attributes

A D E

Questão 13 - Falso

SWISH

File Edit Examples Help

188 users online

Search

Program

```
523 >split(X,L1,L2);
524 mergesort(L1,M1),
525 mergesort(L2,M2),
526 merge(M1,M2,Z).
527
528 split([],[],[]) :- !.
529 split([X],X,[]) :- !.
530 split([X,Y|Z],X|L1,[Y|L2]) :-
531   split(Z,L1,L2).
532
533 %*****
534
535 schema([propertyid, countyname, lotno, area, price, taxrate]).
536 fds([[[propertyid], [countyname, lotno, area, price, taxrate]],
537 [[countyname, lotno], [propertyid, area, price, taxrate]],
538 [[countyname], [taxrate]],
539 [[area], [price]]]
540 ).
541 decomp([propertyid, area, lotno]).
542 decomp([area, countyname]).
543 decomp([area, price]).
544 decomp([countyname, taxrate]).
545
```

procedure 'schema(A)' does not exist

schema(R), fds(F), decomp(D), ljd(R,F,D).

[[b, 1, 1], [b, 1, 2], [b, 1, 3], [b, 1, 4], [b, 1, 5], [b, 1, 6]]
[[b, 2, 1], [b, 2, 2], [b, 2, 3], [b, 2, 4], [b, 2, 5], [b, 2, 6]]
[[b, 3, 1], [b, 3, 2], [b, 3, 3], [b, 3, 4], [b, 3, 5], [b, 3, 6]]

[[b, 1, 1], [b, 1, 2], [b, 1, 3], [b, 1, 4], [b, 1, 5], [b, 1, 6]]
[[b, 2, 1], [b, 2, 2], [b, 2, 3], [b, 2, 4], [b, 2, 5], [b, 2, 6]]

[[b, 1, 1], [b, 1, 2], [b, 1, 3], [b, 1, 4], [b, 1, 5], [b, 1, 6]]
[[b, 2, 1], [b, 2, 2], [b, 2, 3], [b, 2, 4], [b, 2, 5], [b, 2, 6]]

[[b, 1, 1], [b, 1, 2], [b, 1, 3], [b, 1, 4], [b, 1, 5], [b, 1, 6]]
[[b, 2, 1], [b, 2, 2], [b, 2, 3], [b, 2, 4], [b, 2, 5], [b, 2, 6]]

false

?- schema(R), fds(F), decomp(D), ljd(R,F,D).

Examples History Solutions

table results Run

questao01(bruno)....nrm questao02(bruno).nrm questao01(bruno)....nrm

Exibir todos

Questão 14

parte 1

The screenshot shows the SWISH Prolog environment. The left pane contains a Prolog program with the following code:

```
1 %*****
2 % xplus computes the closure of a set of attributes with respect to R and F
3 %
4 % xplus(R,F,X,XPLUS) is true iff XPLUS is the closure of X wrt R and F.
5 %
6 % input: R,F, and X
7 % output: XPLUS
8 %*****
9 xplus(R,F,X,XPLUS) :-
10   xph(R,F,X,X,[],XP),
11   mergesort(XP,XPLUS).
12
13 xph(_,_,T,T,T) :- !.
14 xph(R,F,X,Tnew,XPLUS) :-
15   onepass(R,F,X,Tnew,S),
16   xph(R,F,X,S,Tnew,XPLUS).
17
18 onepass(_,[],_,S) :- !.
19 onepass(R,[U,V]|L,X,T,F) :-
20   subset(T,U), !,
21   union(T,V,T1),
22   onepass(R,L,X,T1,F).
23 onepass(R,[_]|L,X,T,F) :-
24   onepass(R,L,X,T,F).
25
26
```

The right pane shows the results of the query `schema(R), fds(F), decomp(D), ljd(R,F,D).`:

```
[[a, 1], [a, 2], [a, 3], [a, 4], [a, 5], [a, 6], [a, 7], [a, 8], [a, 9], [a, 10]]
[[a, 1], [b, 2], [b, 2, 3], [a, 4], [a, 5], [b, 2, 6], [b, 2, 7], [b, 2, 8], [a, 9], [a, 10]]
[[b, 3, 1], [a, 2], [b, 3, 3], [b, 3, 4], [b, 3, 5], [a, 6], [a, 7], [a, 8], [b, 3, 9], [b, 3, 10]]
[[b, 4, 1], [b, 4, 2], [b, 4, 3], [b, 4, 4], [b, 4, 5], [a, 6], [a, 7], [a, 8], [b, 4, 9], [b, 4, 10]]
[[b, 5, 1], [b, 5, 2], [b, 5, 3], [a, 4], [b, 5, 5], [b, 5, 6], [b, 5, 7], [b, 5, 8], [a, 9], [a, 10]]

D = [[a, b, c], [a, d, e], [b, f, g, h], [d, i, j]].
F = [[a, b], [c], [a], [d, e]], [[b], [f]], [[f], [g, h]], [d], [i, j]].
R = [a, b, c, d, e, f, g, h, i, j].
```

Buttons for 'Next', '10', '100', '1,000', and 'Stop' are visible. Below the results, there is a section for the query `?- schema(R), fds(F), decomp(D), ljd(R,F,D).` with buttons for 'Examples', 'History', 'Solutions', and a 'Run!' button.

parte 2

The screenshot shows the SWISH Prolog environment. The left pane contains a Prolog program with the following code:

```
524 mergesort(L1,M2),
525 merge(M1,M2,Z).
526
527 split([],[],[]) :- !.
528 split([X],[],[]) :- !.
529 split([X,Y|Z],[X|L1],[Y|L2]) :-
530   split(Z,L1,L2).
531
532 %*****
533 %
534 %
535 schema([a,b,c,d,e,f,g,h,i,j]).
536 fds([[[a,b],[c]], [[a],[d,e]], [[b],[f]], [[f],[g,h]], [[d],[i,j]]]).
537 decomp([[[a,b,c,d,e],[b,f,g,h], [d,i,j]]]).
538
539 % Use as Linhas abaixo na janela do SWI-prolog para obter a resposta
540 % schema(R), fds(F), decomp(D), ljd(R,F,D).
541 % schema(R), fds(F), decomp(D), fpd(R,F,D).
542
543 % Ou descomente a linha abaixo para usar answer1 e answer2 para ver a resposta
544 % answer1 :- schema(R), fds(F), decomp(D), ljd(R,F,D).
545 % answer2 :- schema(R), fds(F), decomp(D), fpd(R,F,D).
546
547
548
```

The right pane shows the results of the query `schema(R), fds(F), decomp(D), ljd(R,F,D).`:

```
[[a, 1], [a, 2], [a, 3], [a, 4], [a, 5], [a, 6], [a, 7], [a, 8], [a, 9], [a, 10]]
[[b, 2, 1], [a, 2], [b, 2, 3], [b, 2, 4], [b, 2, 5], [a, 6], [a, 7], [a, 8], [b, 2, 9], [b, 2, 10]]
[[b, 3, 1], [b, 3, 2], [b, 3, 3], [a, 4], [b, 3, 5], [b, 3, 6], [b, 3, 7], [b, 3, 8], [a, 9], [a, 10]]

D = [[a, b, c, d, e], [b, f, g, h], [d, i, j]].
F = [[a, b], [c], [a], [d, e]], [[b], [f]], [[f], [g, h]], [d], [i, j]].
R = [a, b, c, d, e, f, g, h, i, j].
```

Buttons for 'Next', '10', '100', '1,000', and 'Stop' are visible. Below the results, there is a section for the query `?- schema(R), fds(F), decomp(D), ljd(R,F,D).` with buttons for 'Examples', 'History', 'Solutions', and a 'Run!' button.

Questão 15

chaves Principais

Find Candidate Keys

Candidate Keys Found

• **m** **y**

candidate keys

Checar Formas Normalizações



2NF

The table is not in 2NF.



3NF

The table is not in 3NF.



BCNF

The table is not in BCNF.

Checar relação BCNF

Attributes

m y p

Functional Dependencies

m y → p

Attributes

mf c

Functional Dependencies

mf → c

Attributes

mf c

Functional Dependencies

mf → c

Attributes

m mf

Functional Dependencies

m → mf

Teste de Decomposição de Perdas

The screenshot shows the SWISH web interface. The top bar includes the SWISH logo, navigation links (File, Edit, Examples, Help), a search bar, and a user status indicator (84 users online). The main editor area displays a Prolog program with the following code:

```

524 merge_sort(L2,M2),
525 merge(M1,M2,Z).
526
527 split([],[],[]) :- !.
528 split([X],[],[]) :- !.
529 split([X,Y|Z],[X|L1],[Y|L2]) :-
530   split(Z,L1,L2).
531
532 %*****
533 %*****
534
535 schema([m, y, p, mf, c]).
536 fds([[m],[m,f]], [[m,y],[p]], [[m,f],[c]]].
537 decomp([[m,y,p], [m, mp, c]]).
538
539 % Use as linhas abaixo na janela do SWI-prolog para obter a resposta
540 % schema(R), fds(F), decomp(D), ljd(R,F,D).
541 % schema(R), fds(F), decomp(D), fpd(R,F,D).
542
543 % OU descomente a linha abaixo para usar answer1 e answer2 para ver a resposta
544 % answer1 :- schema(R), fds(F), decomp(D), ljd(R,F,D).
545 % answer2 :- schema(R), fds(F), decomp(D), fpd(R,F,D).
546
547
548

```

Below the code, there is a query window showing the execution of the query:

```

?- schema(R), fds(F), decomp(D), ljd(R,F,D).
false

```

The background of the interface features a large, stylized owl illustration.