Welcome To BigBlueButton

BigBlueButton is an open source web conferencing system designed for online learning

CHAT

Send public and private messages.

WEBCAMS

Hold visual meetings.

AUDIO

Communicate using high quality audio.

EMOJIS

Express yourself.

BREAKOUT ROOMS

Group users into breakout rooms for team collaboration.

POLLING

Poll your users anytime.

SCREEN SHARING

Share your screen.

MULTI-USER WHITEBOARD

Draw together.

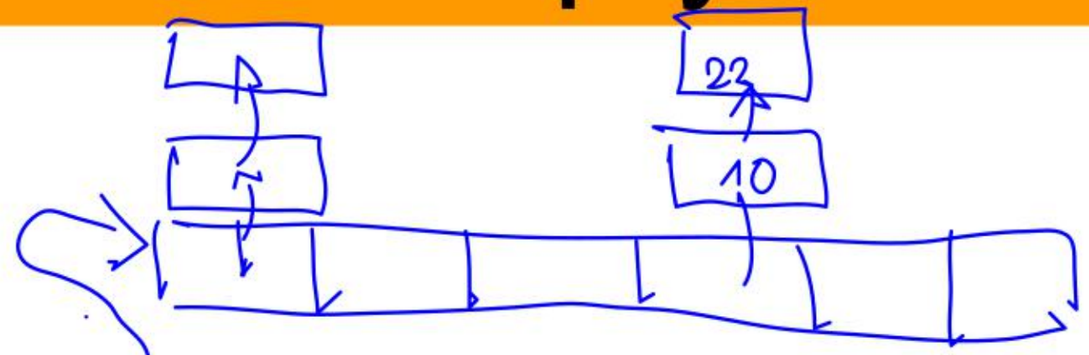
For more information visit bigbluebutton.org →

Vyhledávání, zejména rozptylování

Rozptylování (hashing)

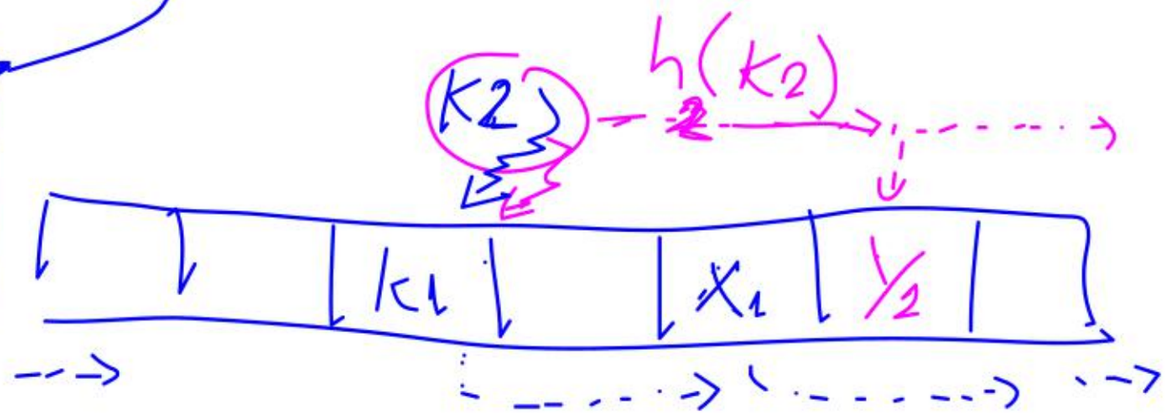
- Rozptylovací funkce
- Řešení kolizí
  - Zřetězené rozptylování
  - Otevřené rozptylování
    - Linear Probing
    - Double hashing

Vyhledávání, zejména rozptylování



Rozptylování (hashing)

- Rozptylovací funkce KLÍČ ↦ ADRESA V TAB
- Řešení kolizí
  - Zřetězené rozptylování
  - Otevřené rozptylování
    - Linear Probing
    - Double hashing



b) Očekávaný počet testů

Linear probing:

Plnění α	1/2	2/3	3/4	9/10
Search hit	1.5	2.0	2.5	5.5
Search miss	2.5	5.0	8.5	50.5

Double hashing:

Plnění α	1/2	2/3	3/4	9/10
Search hit	1.4	1.6	1.8	2.6
Search miss	2.0	3.0	4.0	10.0

Tabulka může být více zaplněná, než začne klesat výkonnost.

K dosažení stejného výkonu stačí menší tabulka.

b) Očekávaný počet testů

Linear probing:

Plnění α	1/2	2/3	3/4	9/10
Search hit	1.5	2.0	2.5	5.5
Search miss	2.5	5.0	8.5	50.5

Double hashing:

Plnění α	1/2	2/3	3/4	9/10
Search hit	1.4	1.6	1.8	2.6
Search miss	2.0	3.0	4.0	10.0

Tabulka může být více zaplněná, než začne klesat výkonnost.

K dosažení stejného výkonu stačí menší tabulka.

ALG 13b

Srůstající hashování

Ukázky

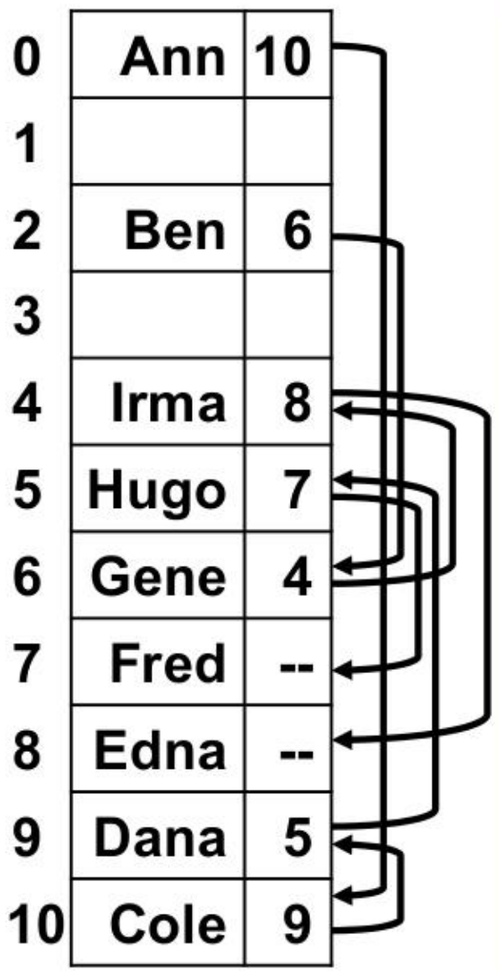
- LISCH (late insert standard coalesced hashing)
- EISCH (early insert standard coalesced hashing)
- LICH (late insert coalesced hashing)
- EICH (early insert coalesced hashing)
- VICH (variable insert coalesced hashing)

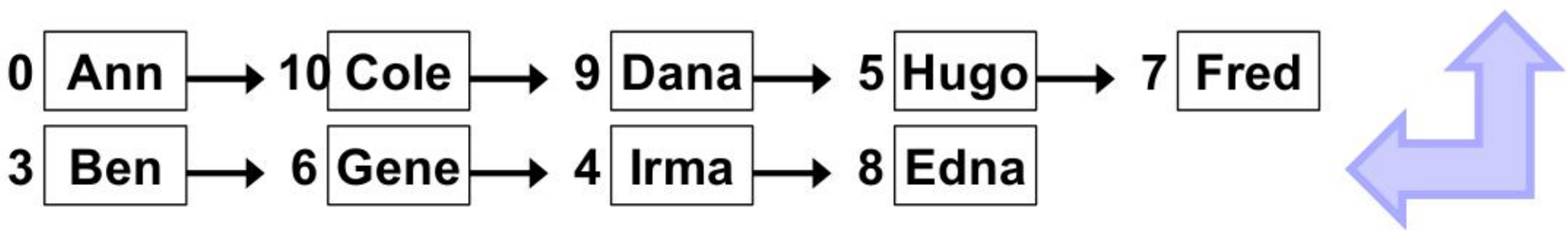
Srůstající hashování -- coalesced hashing

Jde o metodu řešení kolizí, nezáleží na konkrétní podobě hashovací funkce h(k).

Synonyma (po kolizi) se ukádají do jednosměrného spojového seznamu synonym. Všechny seznamy jsou "propleteně" uloženy přímo v tabulce.
Tabulka ke každému klíči obsahuje ukazatel na další klíč v seznamu.
Každý klíč je součástí některého seznamu synonym.

Při vyhledávání se postupuje stejně jako při vkládání, v podstatě jde o lineární prohledávání spojového seznamu.



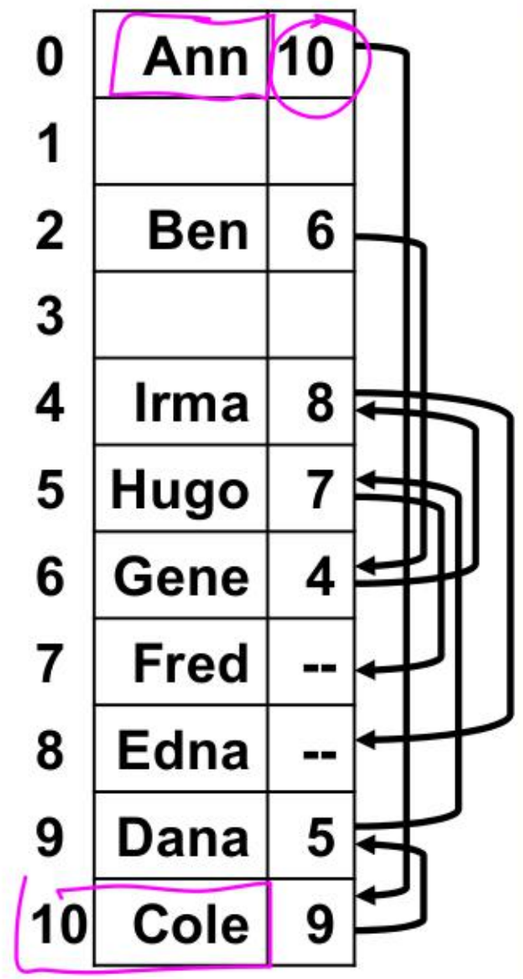


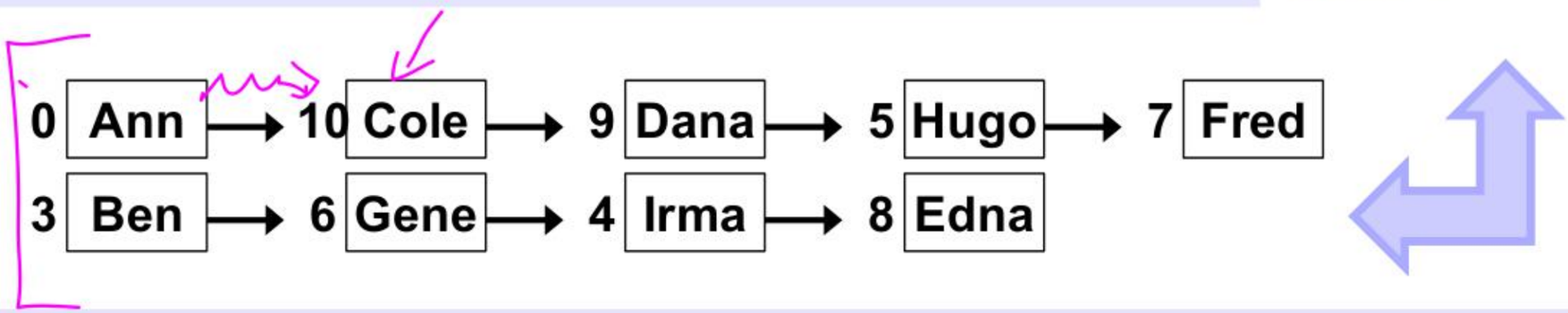
Srůstající hashování -- coalesced hashing

Jde o metodu řešení kolizí, nezáleží na konkrétní podobě hashovací funkce h(k).

Synonyma (po kolizi) se ukádají do jednosměrného spojového seznamu synonym. Všechny seznamy jsou "propleteně" uloženy přímo v tabulce.
Tabulka ke každému klíči obsahuje ukazatel na další klíč v seznamu.
Každý klíč je součástí některého seznamu synonym.

Při vyhledávání se postupuje stejně jako při vkládání, v podstatě jde o lineární prohledávání spojového seznamu.





LISCH (late insert standard coalesced hashing)

Hashovací funkce h, data d.
Pozice p := h(d);
Prohledej seznam začínající na pozici p a pokud nenajdeš d, přidej d do tabulky na první volné místo od konce tabulky a připoj ho do seznamu synonym d na poslední místo.

Ukazatel na první volné místo od konce tabulky. Po každém přidání prvku se aktualizuje.

	Name	Next
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	8	7

LISCH (late insert standard coalesced hashing)

Hashovací funkce h, data d.
Pozice p := h(d);
Prohledej seznam začínající na pozici p a pokud nenajdeš d, přidej d do tabulky na první volné místo od konce tabulky a připoj ho do seznamu synonym d na poslední místo.

Ukazatel na první volné místo od konce tabulky. Po každém přidání prvku se aktualizuje.

	Name	Next
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	8	7

LISCH (late insert standard coalesced hashing)

0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		

0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		

0	Ann	9
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9	Cole	--

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	8	7

LISCH (late insert standard coalesced hashing)

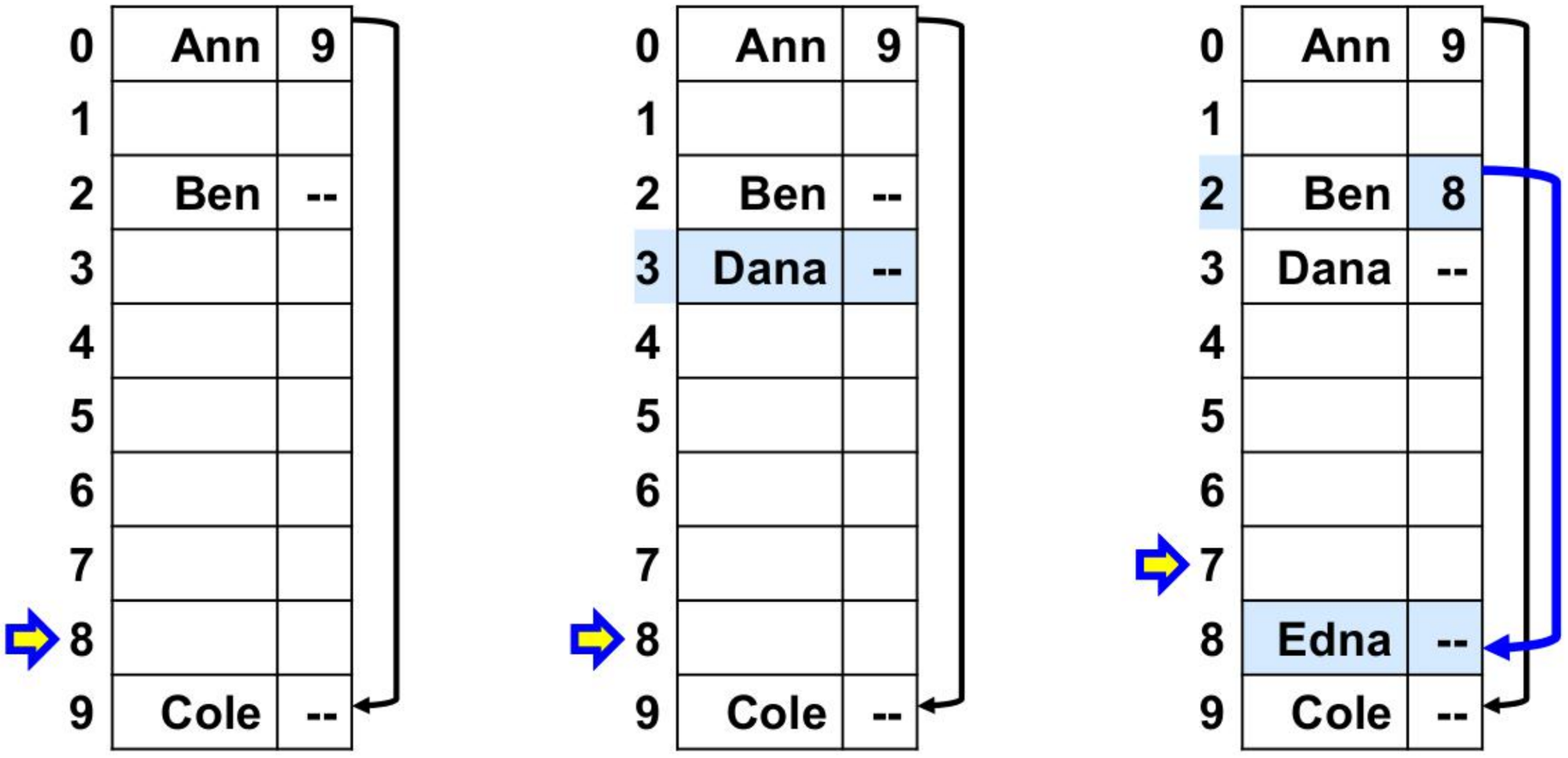
0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		

0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		

0	Ann	9
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9	Cole	--

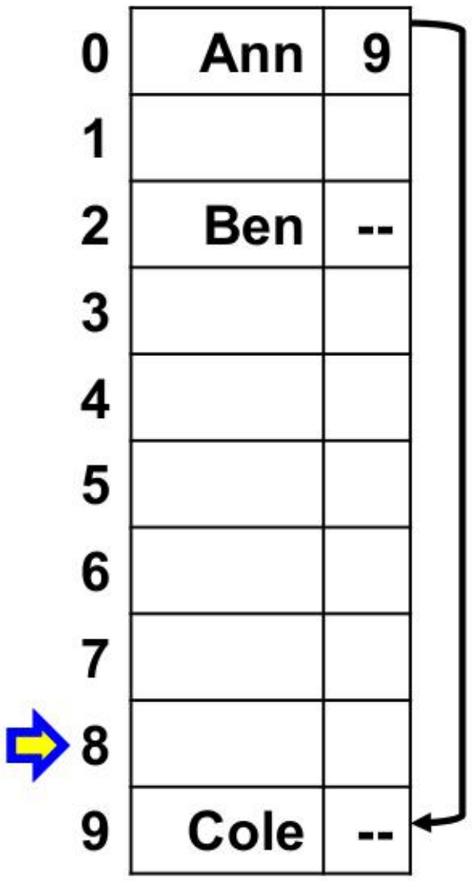
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	8	7

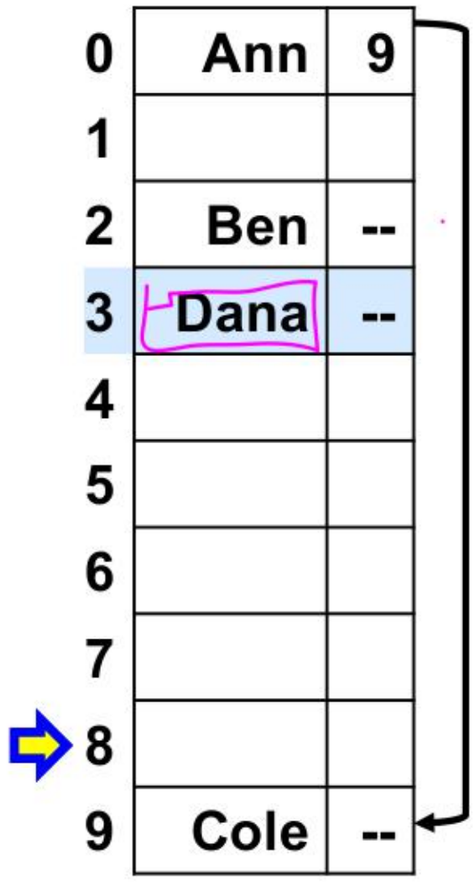
LISCH (late insert standard coalesced hashing)

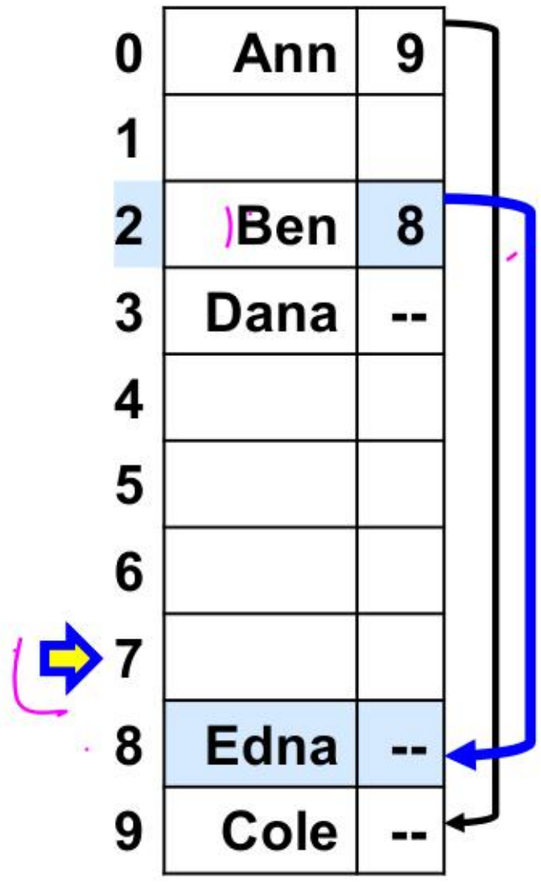


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	8	7

LISCH (late insert standard coalesced hashing)

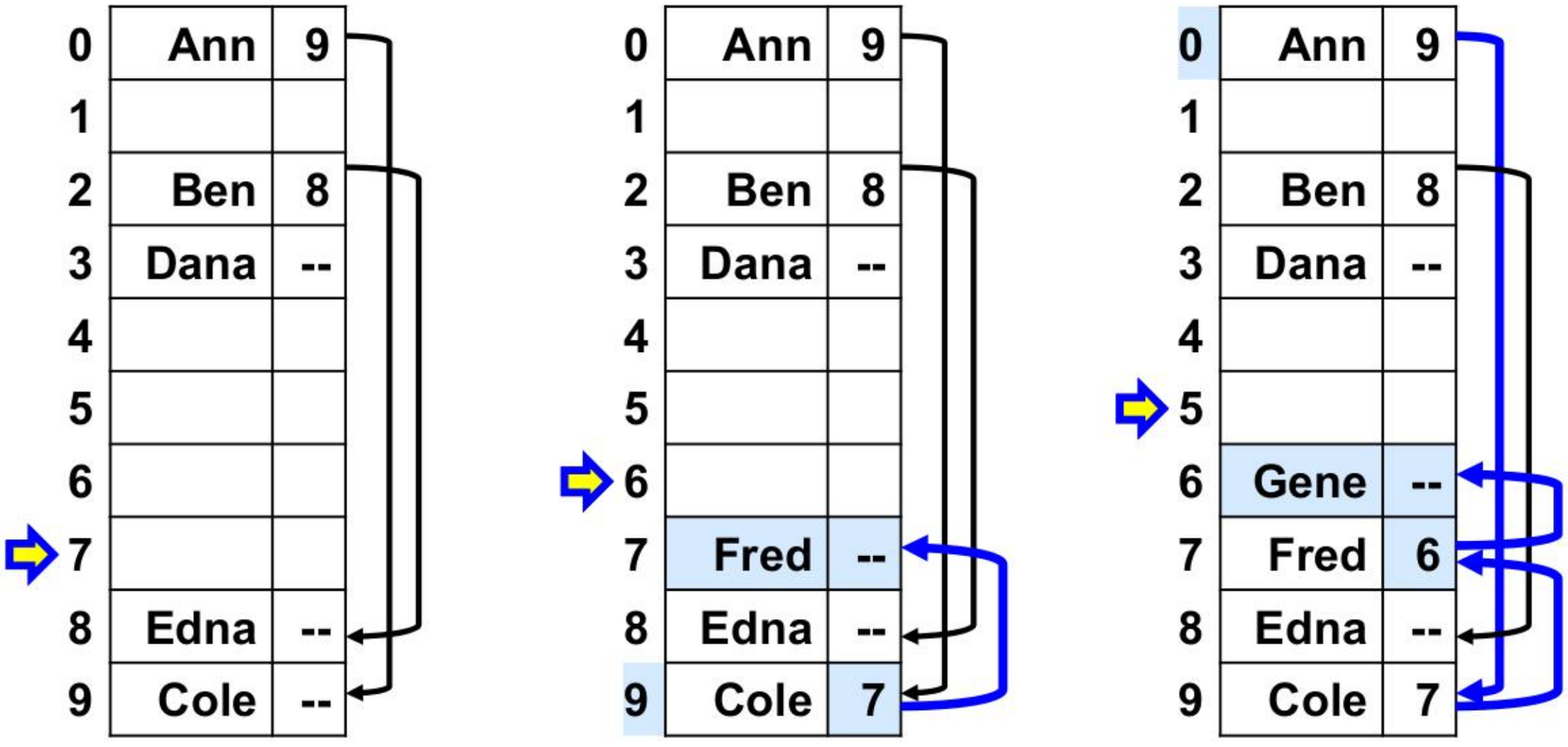






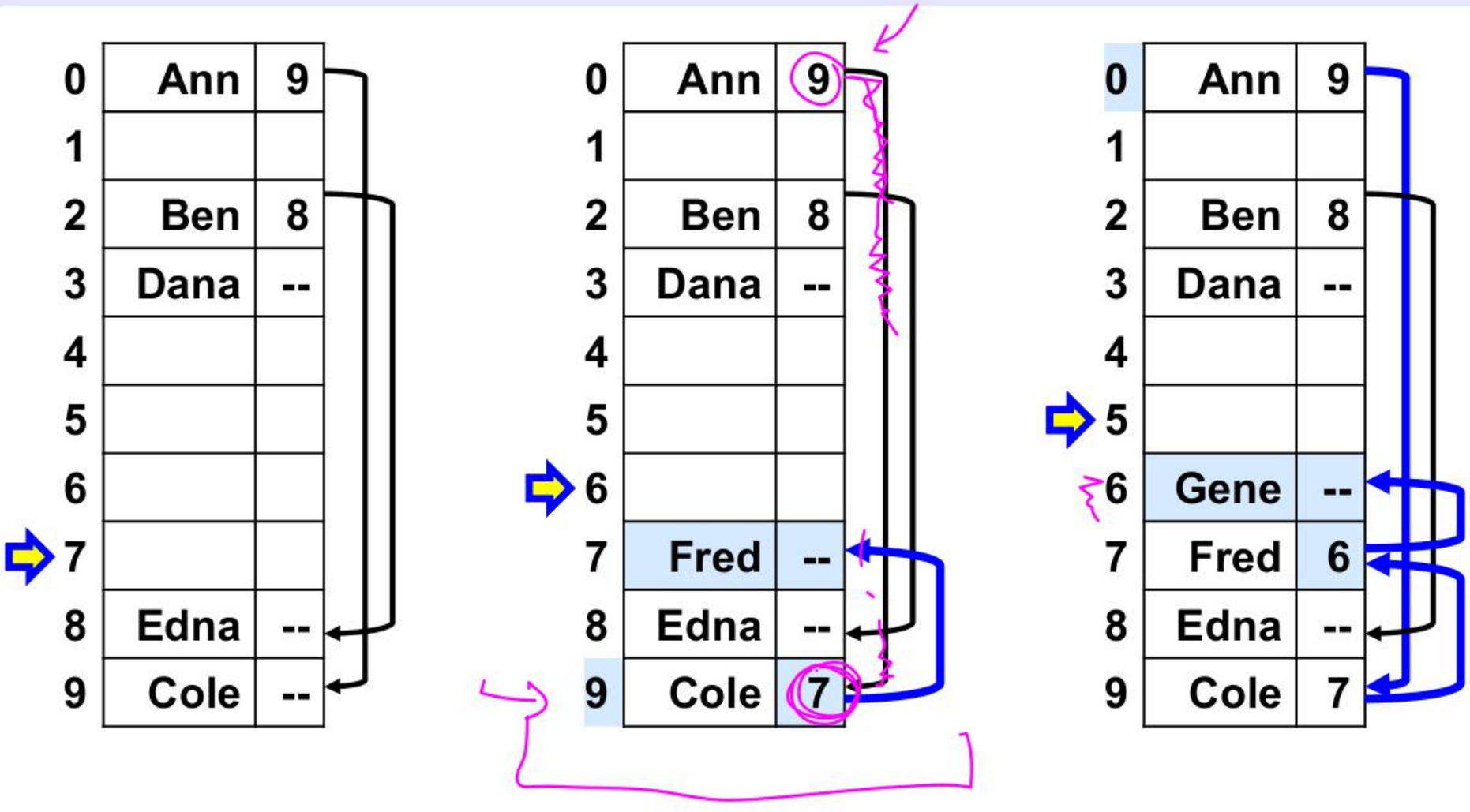
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	8	7

LISCH (late insert standard coalesced hashing)



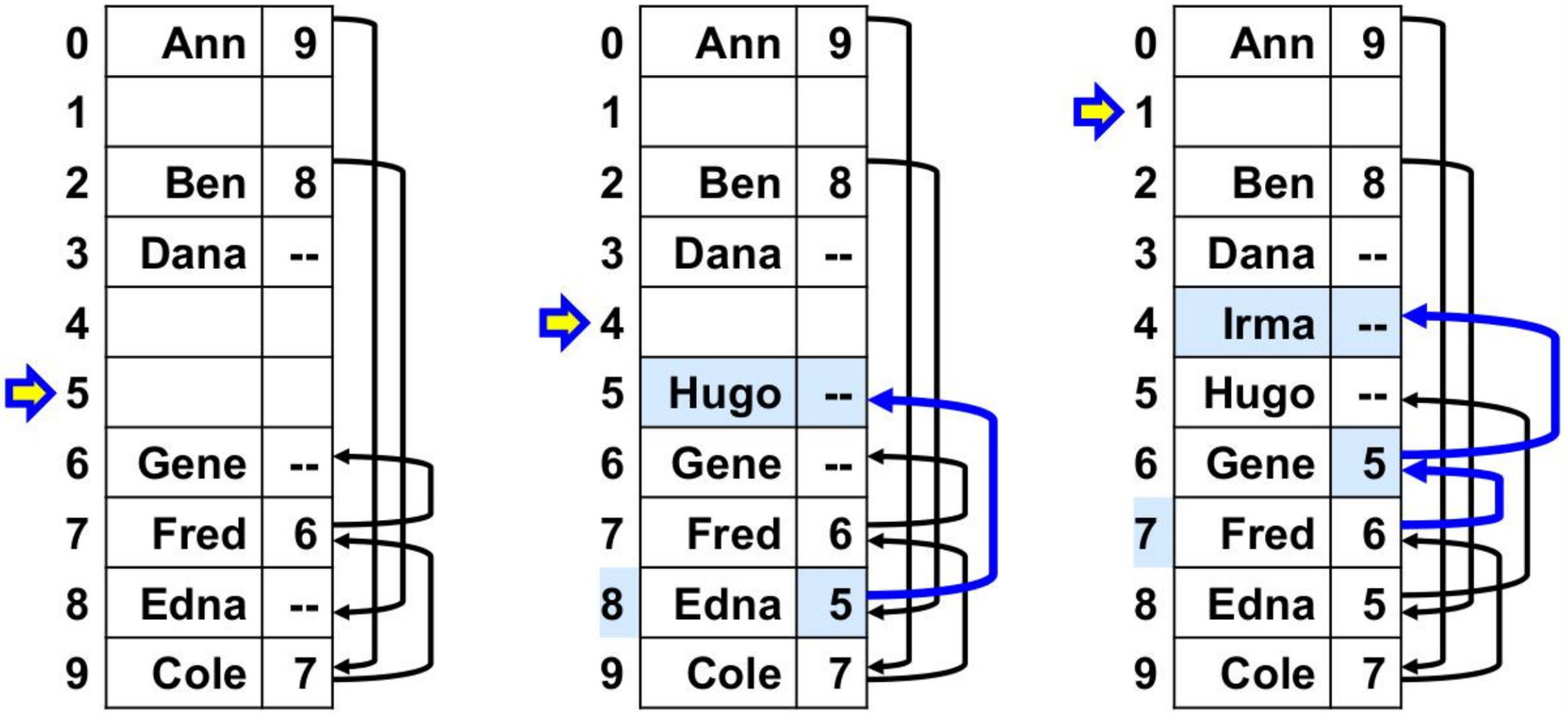
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	8	7

LISCH (late insert standard coalesced hashing)



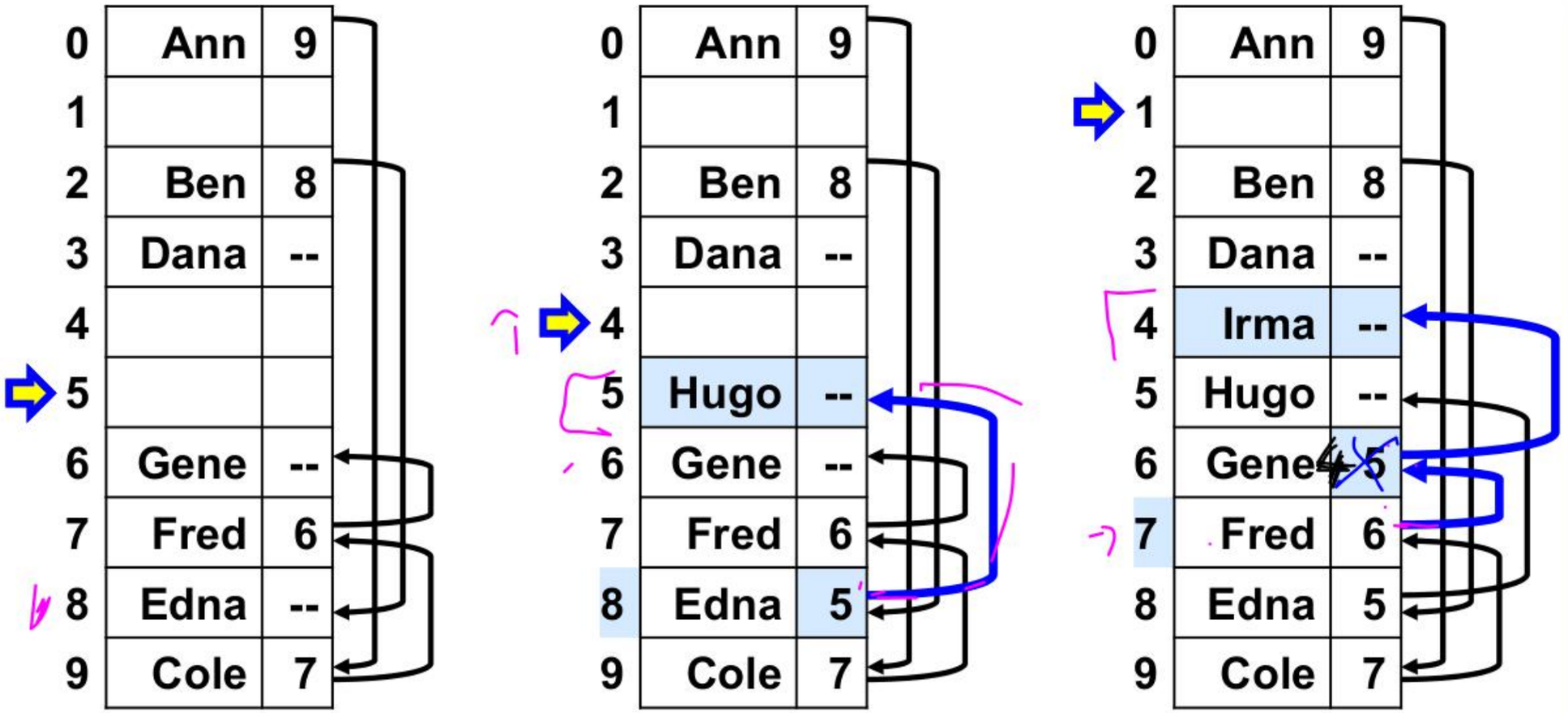
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	8	7

LISCH (late insert standard coalesced hashing)



data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	8	7

LISCH (late insert standard coalesced hashing)



data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	8	7

EISCH (early insert standard coalesced hashing)

Hashovací funkce h, data d.
Pozice p := h(d);
Prohledej seznam začínající na pozici p a pokud nenajdeš d, přidej d do tabulky na první volné místo od konce tabulky a připoj ho do seznamu synonym d za první místo.

Ukazatel na první volné místo od konce tabulky. Po každém přidání prvku se aktualizuje.

	Name	Next
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	2	6

EISCH (early insert standard coalesced hashing)

Hashovací funkce h, data d.
Pozice p := h(d);
Prohledej seznam začínající na pozici p a pokud nenajdeš d, přidej d do tabulky na první volné místo od konce tabulky a připoj ho do seznamu synonym d za první místo.

	Name	Next
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		

Ukazatel na první volné místo od konce tabulky. Po každém přidání prvku se aktualizuje.

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	2	6

EISCH (early insert standard coalesced hashing)

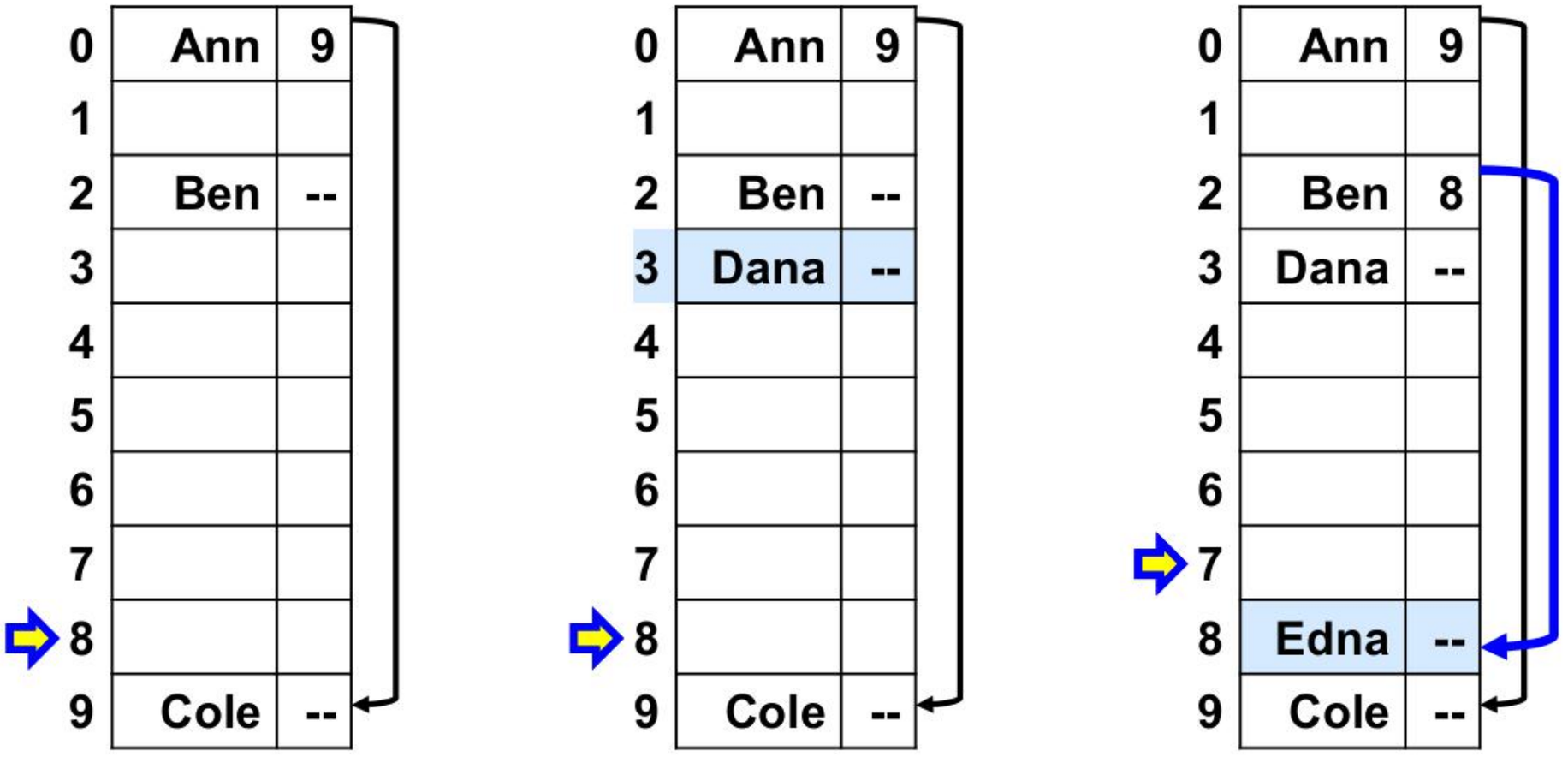
0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		

0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		

0	Ann	9
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9	Cole	--

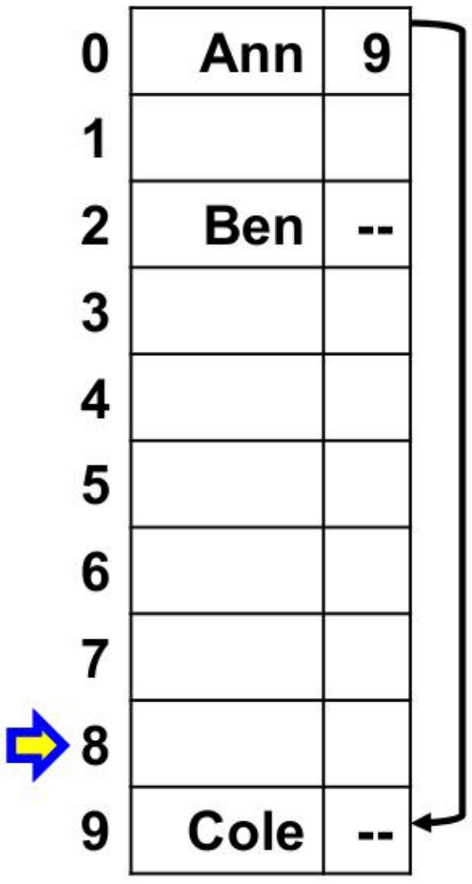
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	2	6

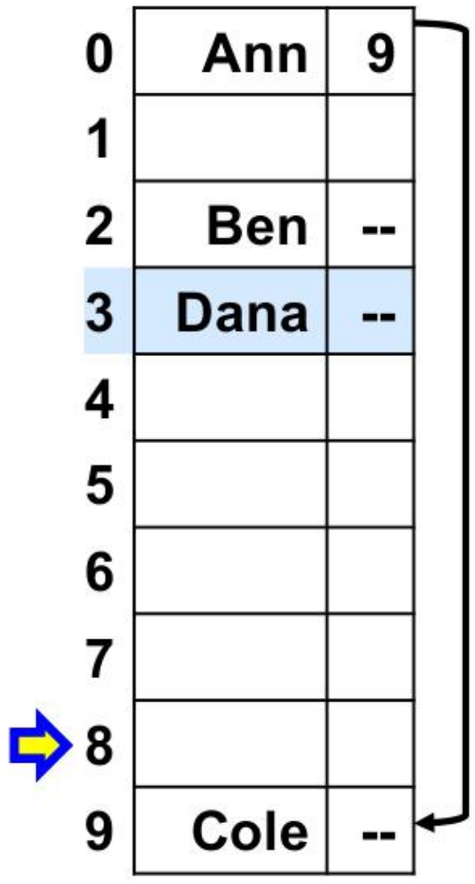
EISCH (early insert standard coalesced hashing)

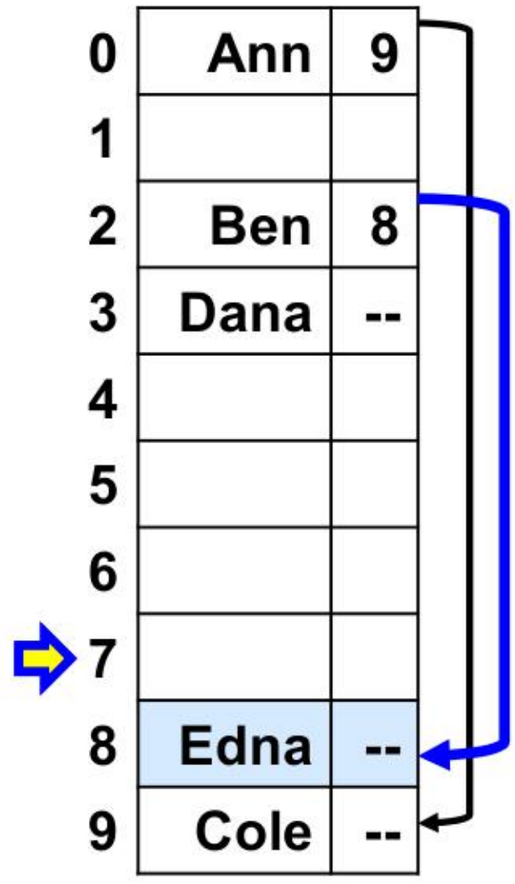


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	2	6

EISCH (early insert standard coalesced hashing)

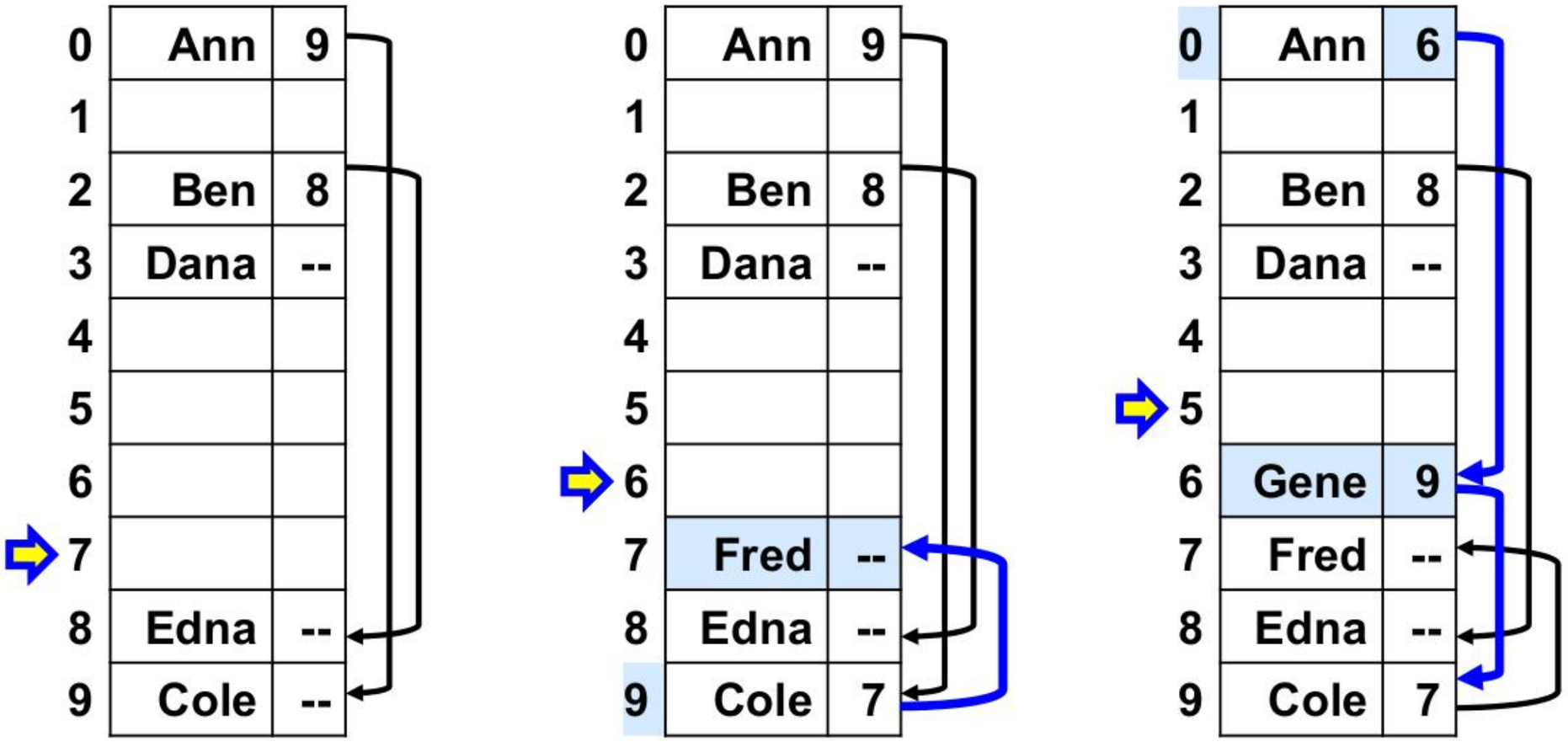






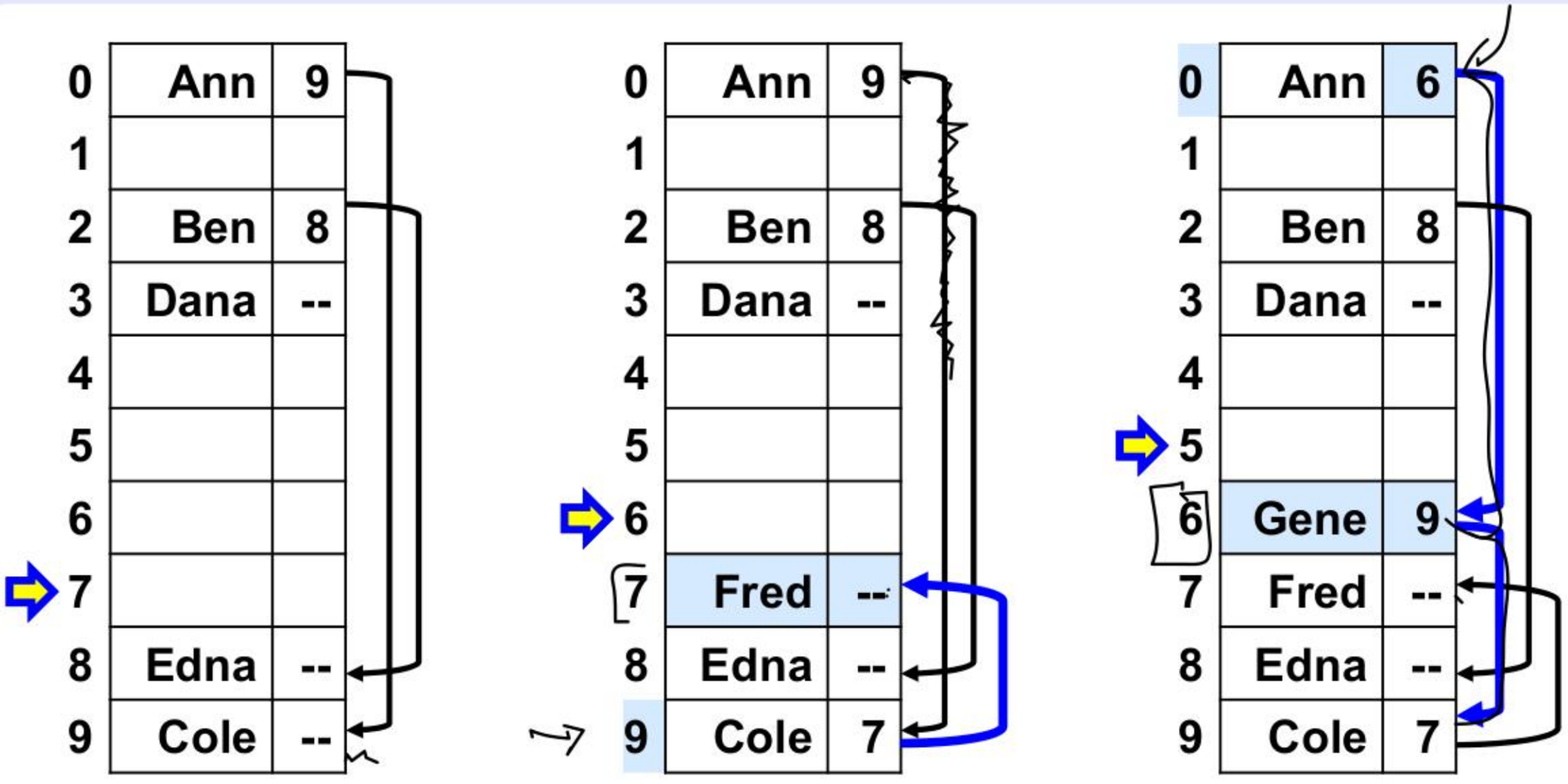
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	2	6

EISCH (early insert standard coalesced hashing)



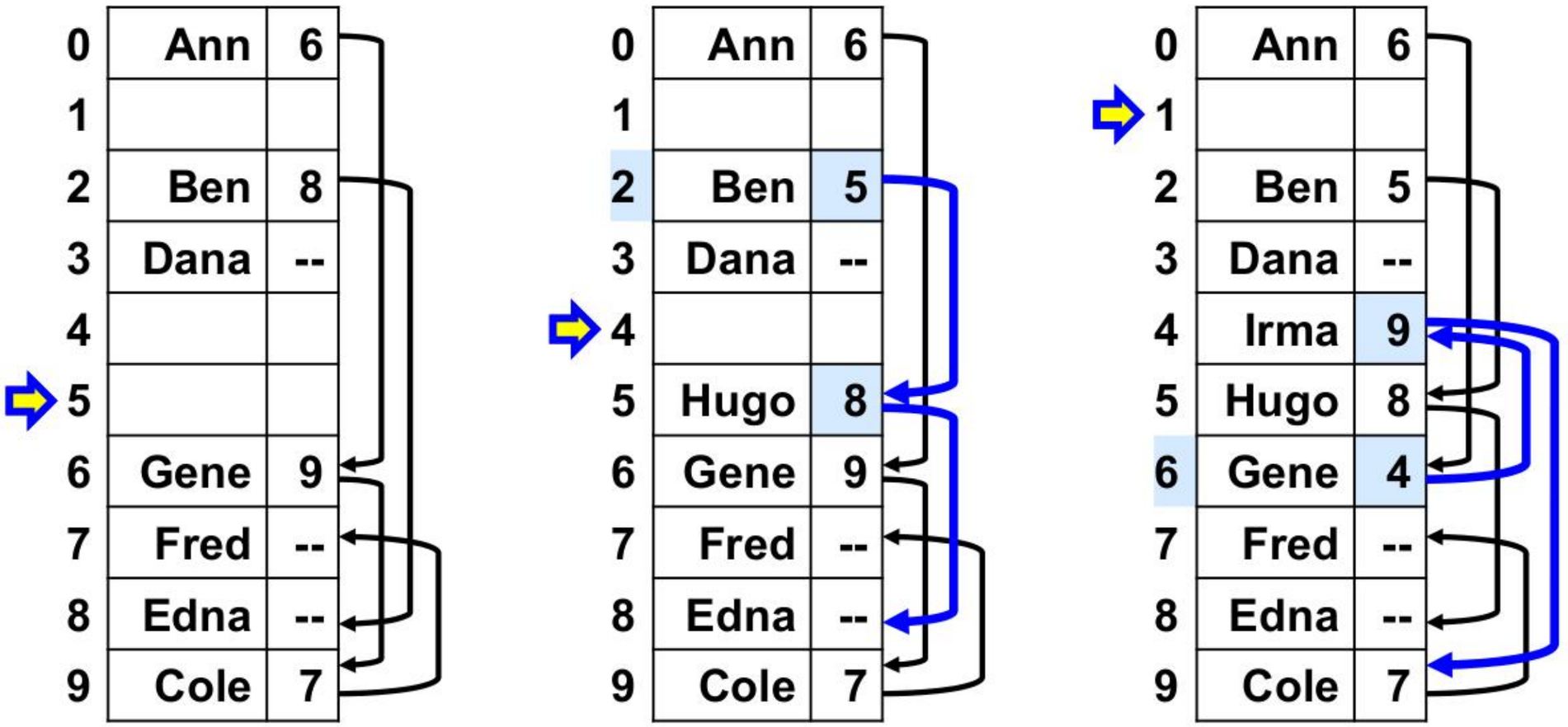
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	2	6

EISCH (early insert standard coalesced hashing)



data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	2	6

EISCH (early insert standard coalesced hashing)



data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	2	6

EISCH (early insert standard coalesced hashing)

0	Ann	6
1		
2	Ben	8
3	Dana	--
4		
5		
6	Gene	9
7	Fred	--
8	Edna	--
9	Cole	7

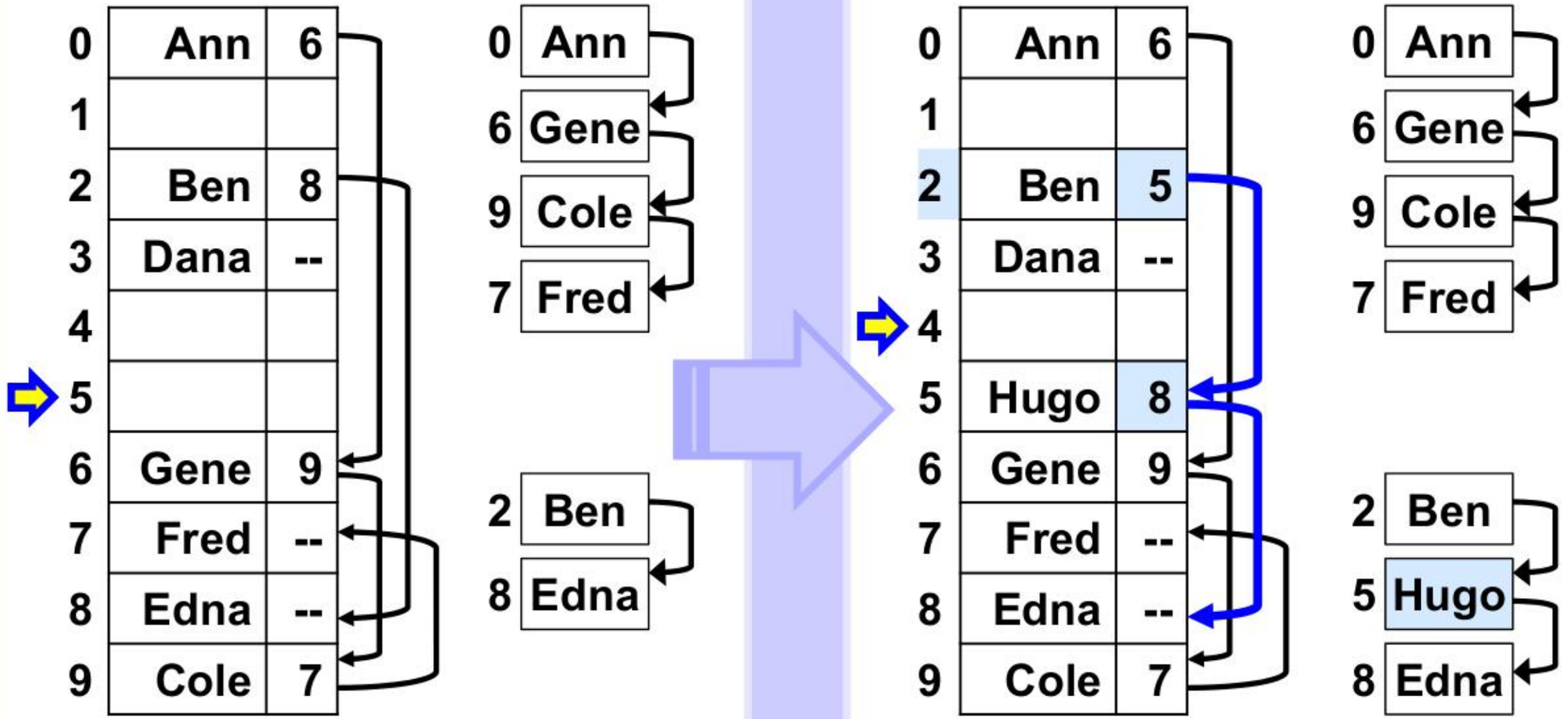
0	Ann	6
1		
2	Ben	5
3	Dana	--
4		
5	Hugo	8
6	Gene	9
7	Fred	--
8	Edna	--
9	Cole	7

0	Ann	6
1		
2	Ben	5
3	Dana	--
4	Irma	9
5	Hugo	8
6	Gene	4
7	Fred	--
8	Edna	--
9	Cole	7

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	2	6

EISCH (early insert standard coalesced hashing)

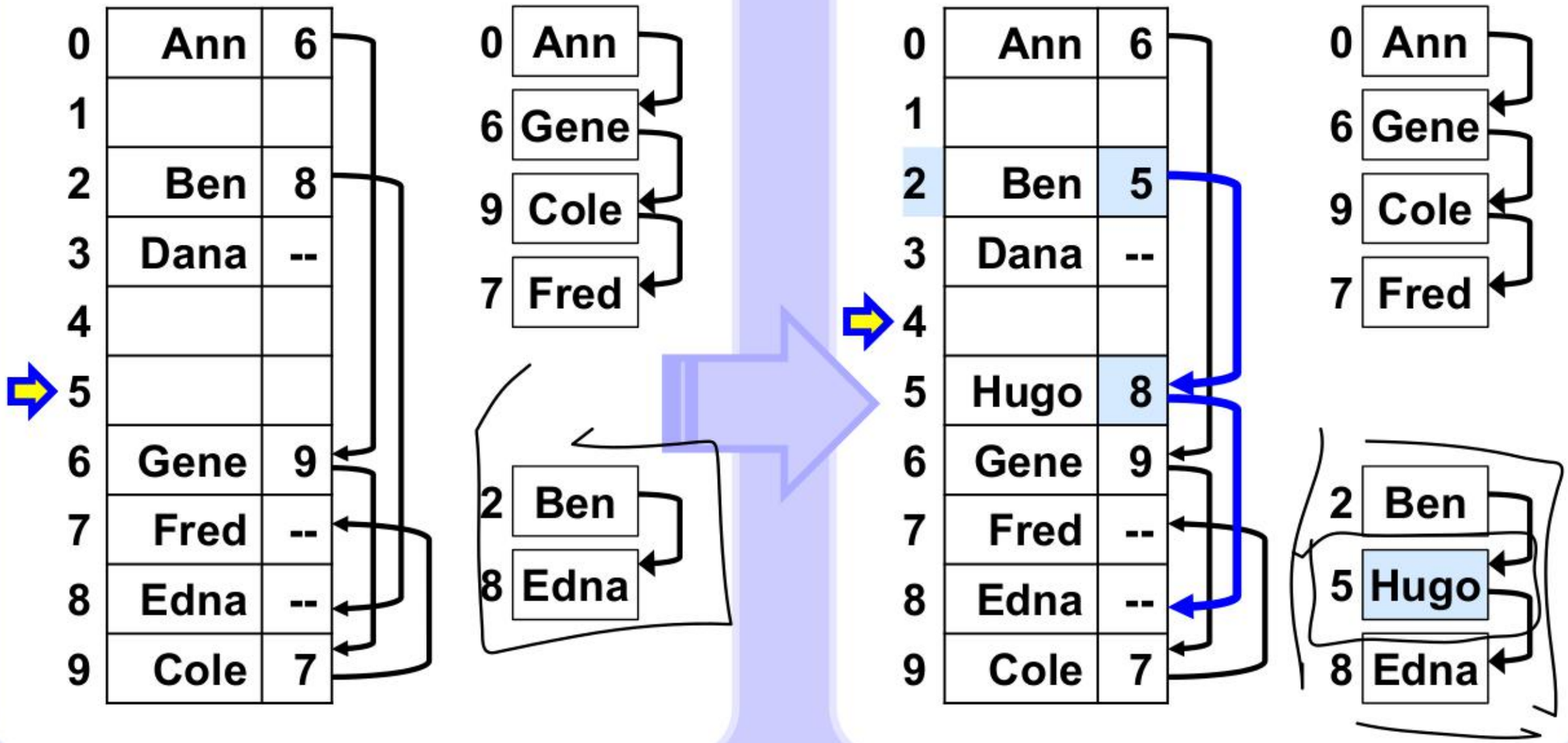
Insert(Hugo)



data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	2	6

EISCH (early insert standard coalesced hashing)

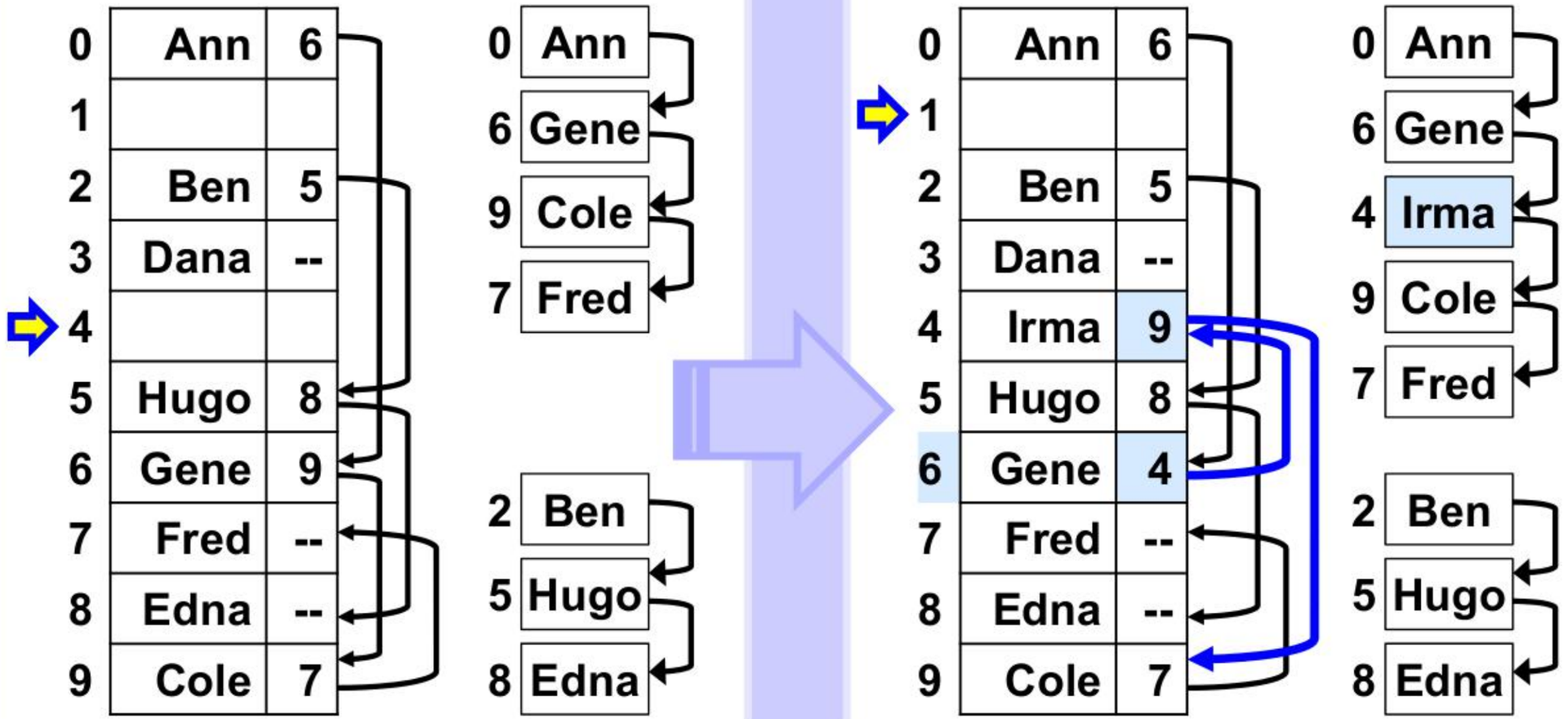
Insert(Hugo)



data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	2	6

EISCH (early insert standard coalesced hashing)

Insert(Irma)



data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	3	2	9	0	2	6

Srůstající hashování s pomocnou pamětí

Pro snížení srůstání a tedy zvýšení efektivity hashování se tabulka rozšiřuje o pomocnou paměť - tzv. sklep (cellar).

Sklep je místo na konci tabulky, které není adresovatelné hashovací funkcí, má ale stejnou strukturu jako celá tabulka.

Algoritmy LICH a EICH jsou analogické varianty algoritmů LISCH a EISCH s přidáním sklepa.
Po naplnění sklepa pokračuje plnění jako v LISCH a EISCH.

Algoritmus VICH (variable insert coalesced hashing) připojuje prvek za poslední prvek seznamu, který je ještě ve sklepě. Pokud ve sklepě žádný není, vkládá jako EISCH, tj. hned za kolidující prvek v seznamu.

Srůstající hashování s pomocnou pamětí

Pro snížení srůstání a tedy zvýšení efektivity hashování se tabulka rozšiřuje o pomocnou paměť - tzv. sklep (cellar).

Sklep je místo na konci tabulky, které není adresovatelné hashovací funkcí, má ale stejnou strukturu jako celá tabulka.

Algoritmy LICH a EICH jsou analogické varianty algoritmů LISCH a EISCH s přidáním sklepa.
Po naplnění sklepa pokračuje plnění jako v LISCH a EISCH.

Algoritmus VICH (variable insert coalesced hashing) připojuje prvek za poslední prvek seznamu, který je ještě ve sklepě. Pokud ve sklepě žádný není, vkládá jako EISCH, tj. hned za kolidující prvek v seznamu.

LICH (late insert coalesced hashing)

0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	10
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10	Cole	--

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)

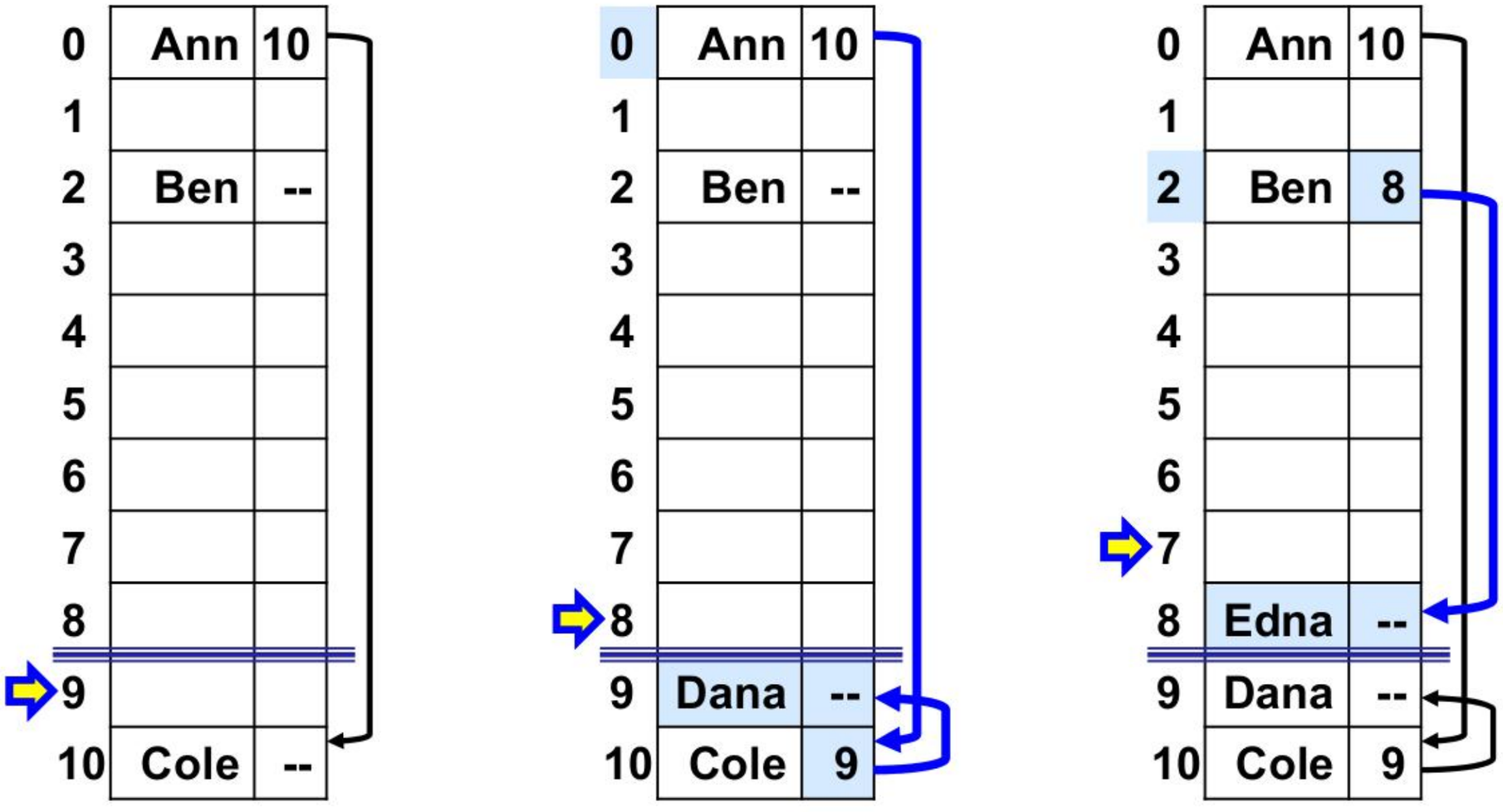
0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	10
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10	Cole	--

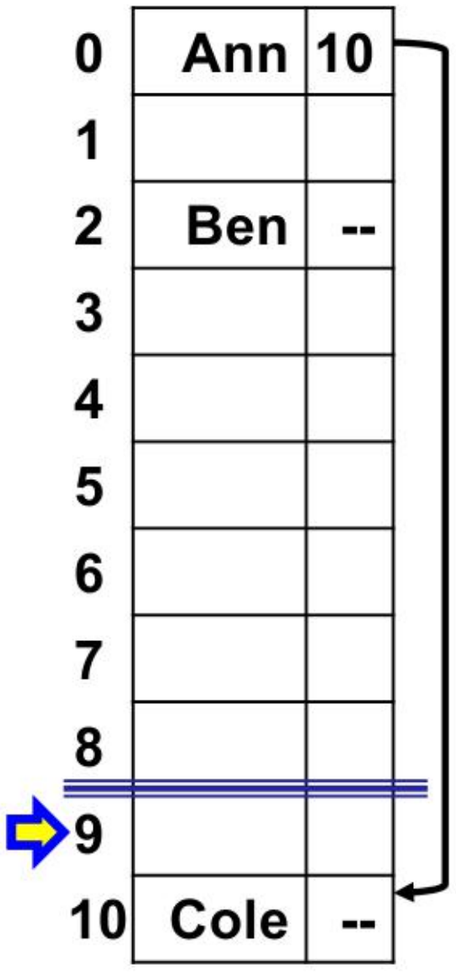
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

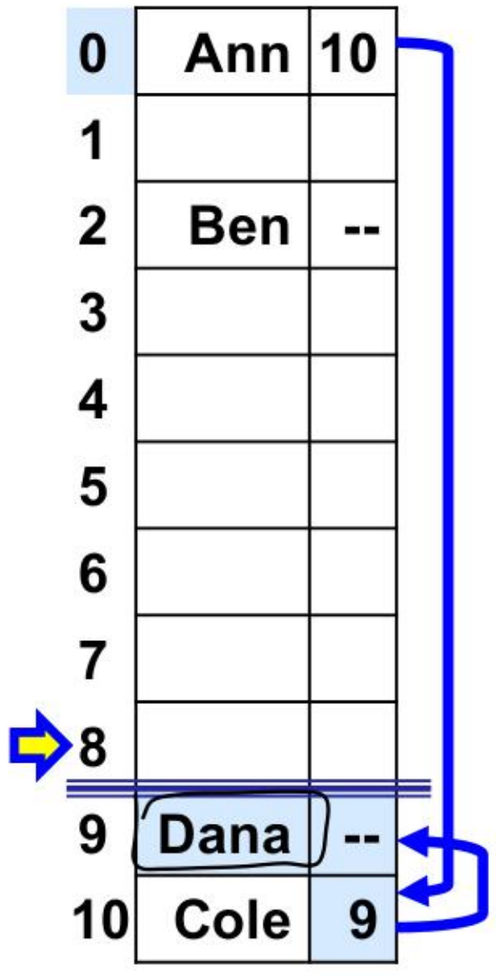
LICH (late insert coalesced hashing)

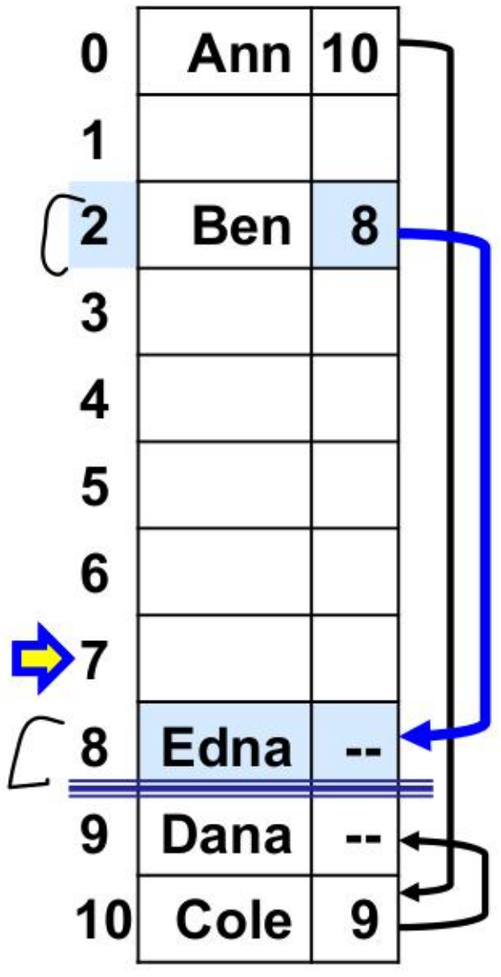


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)

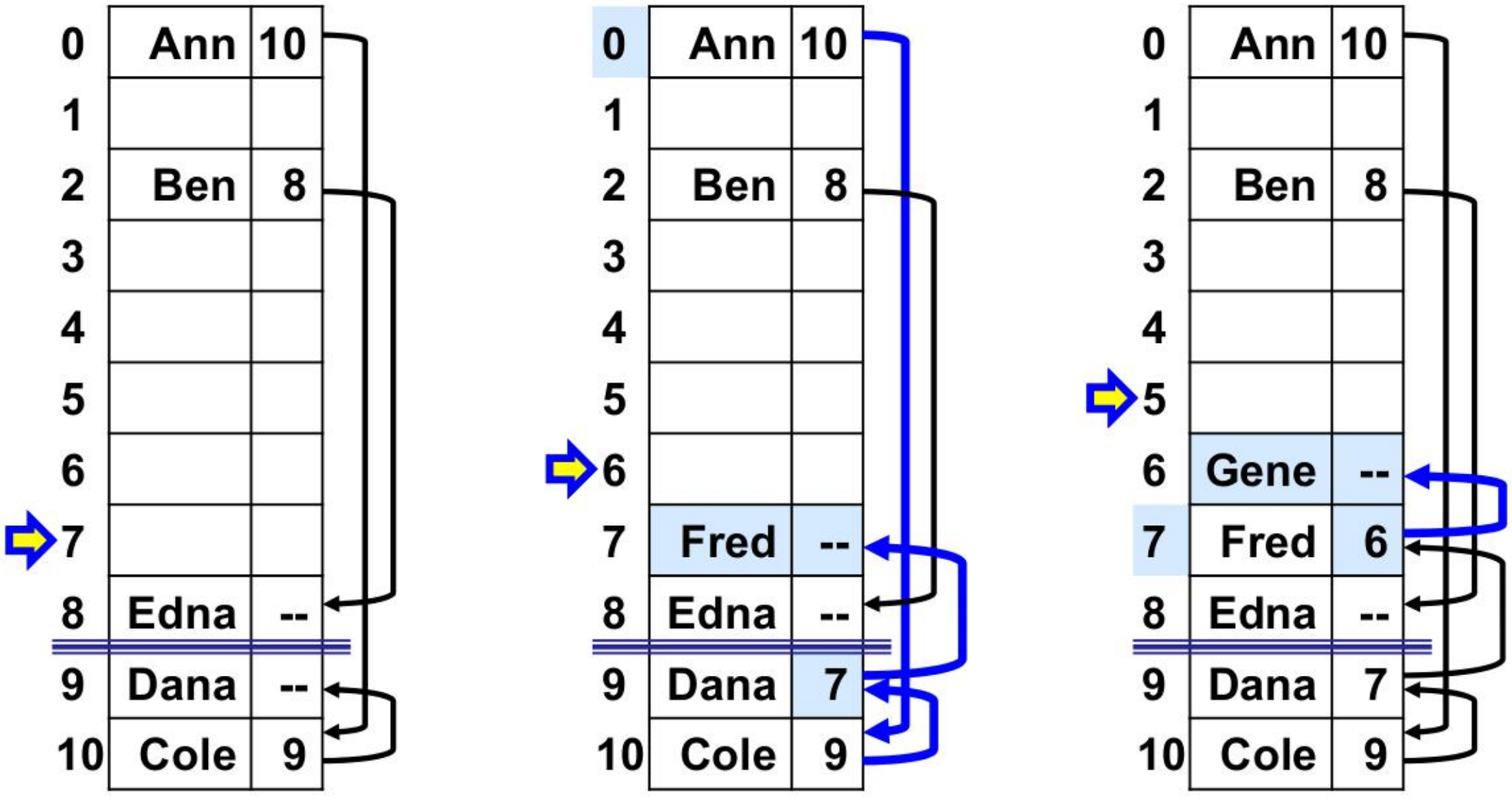






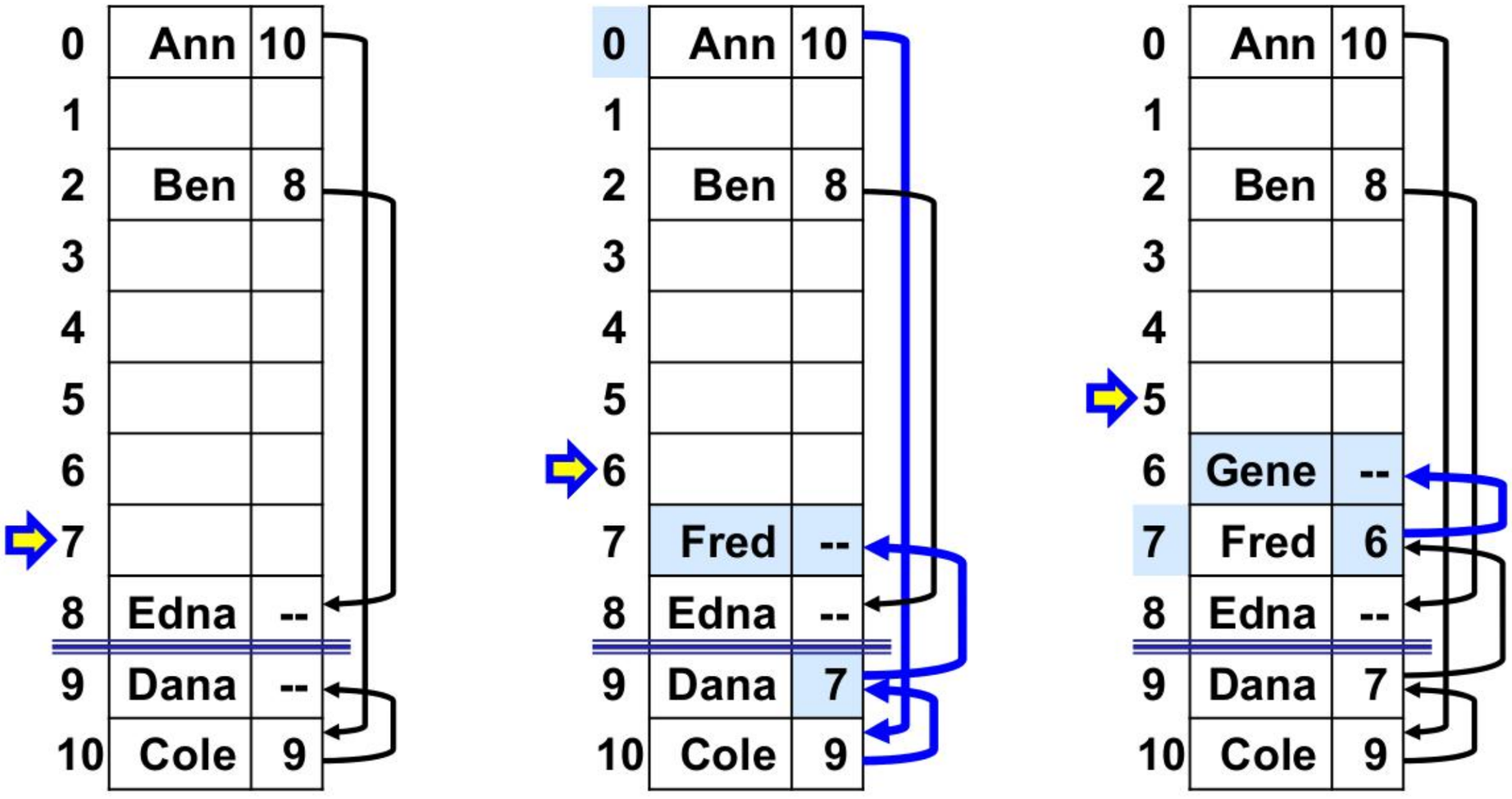
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)



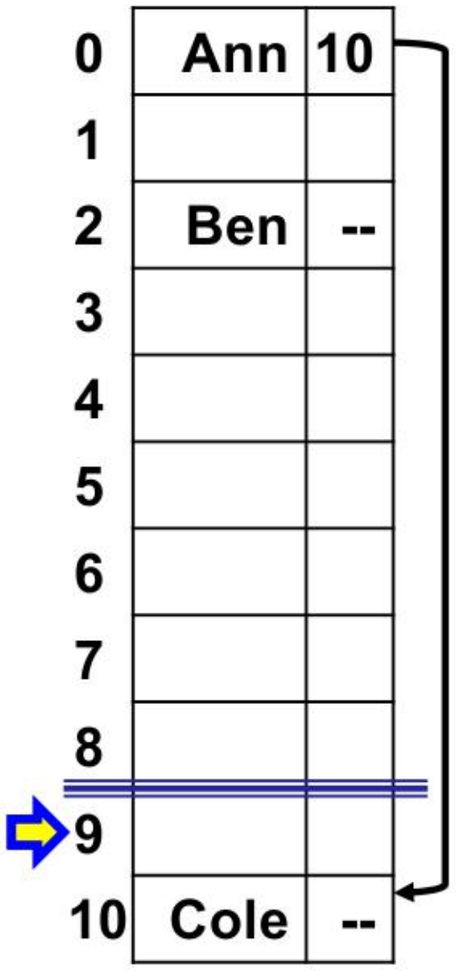
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

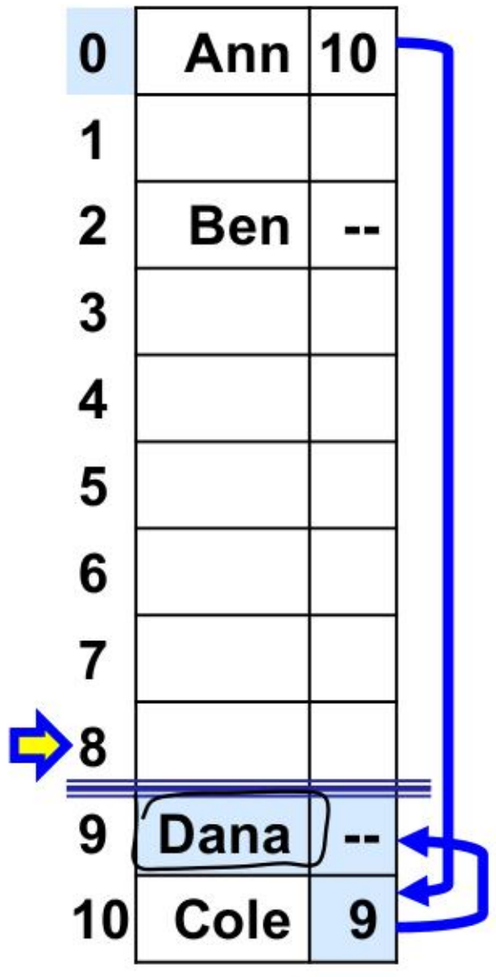
LICH (late insert coalesced hashing)

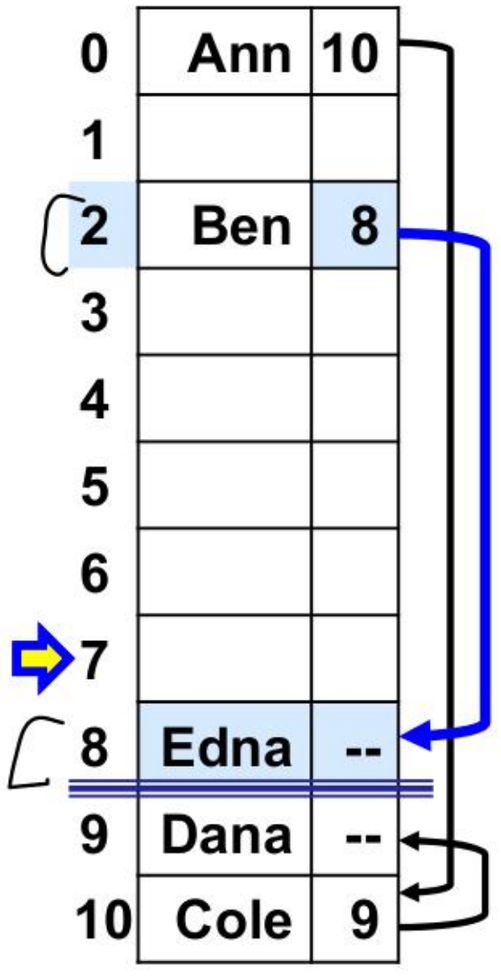


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)

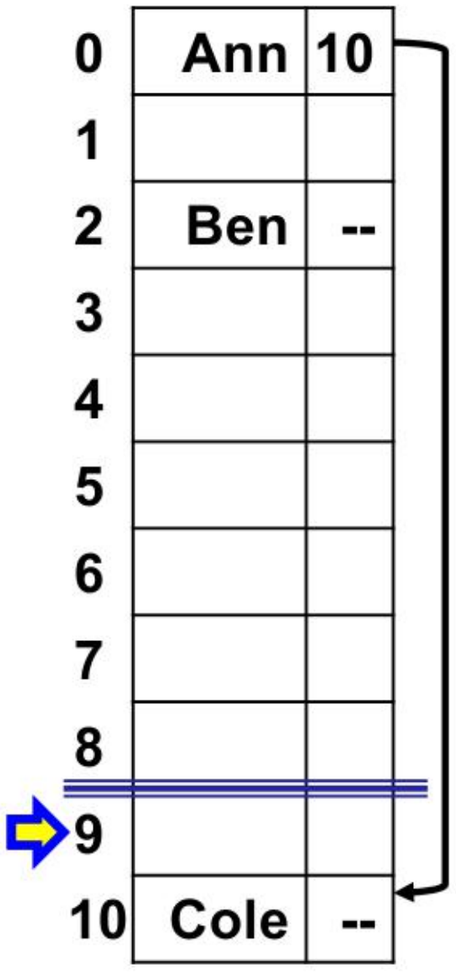


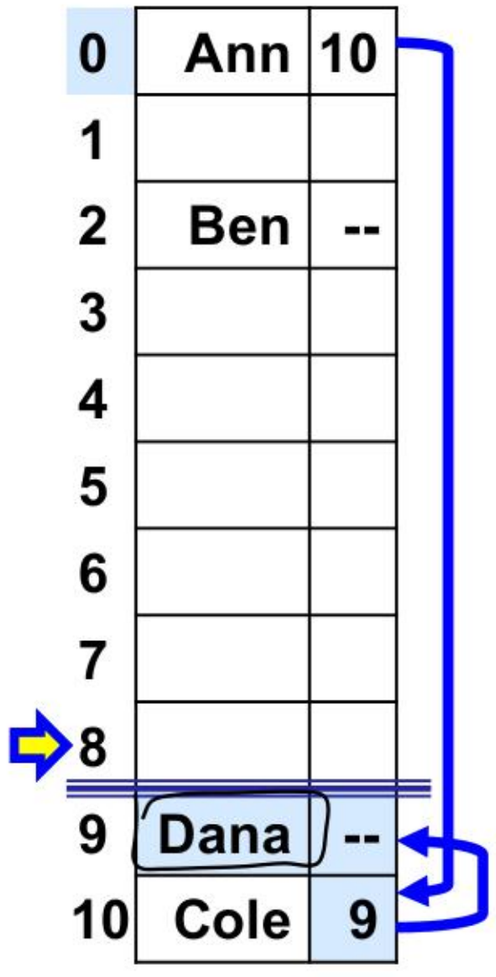


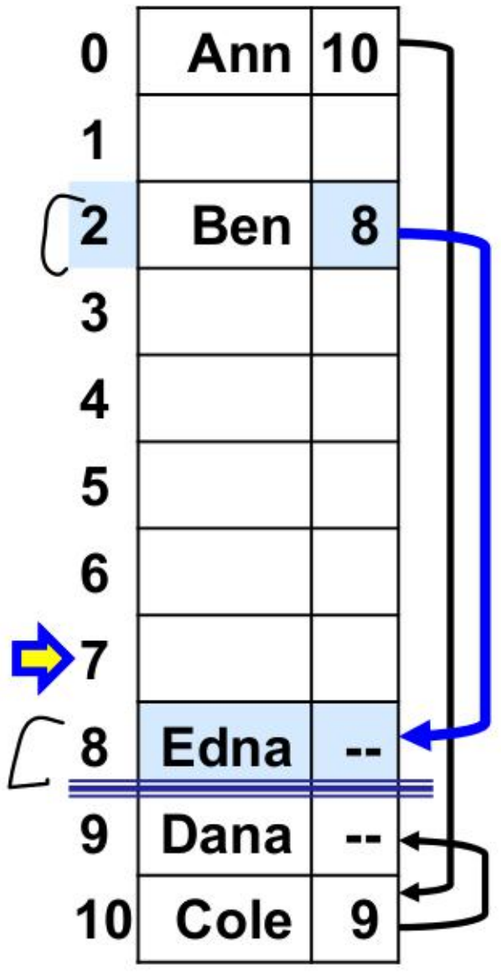


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)

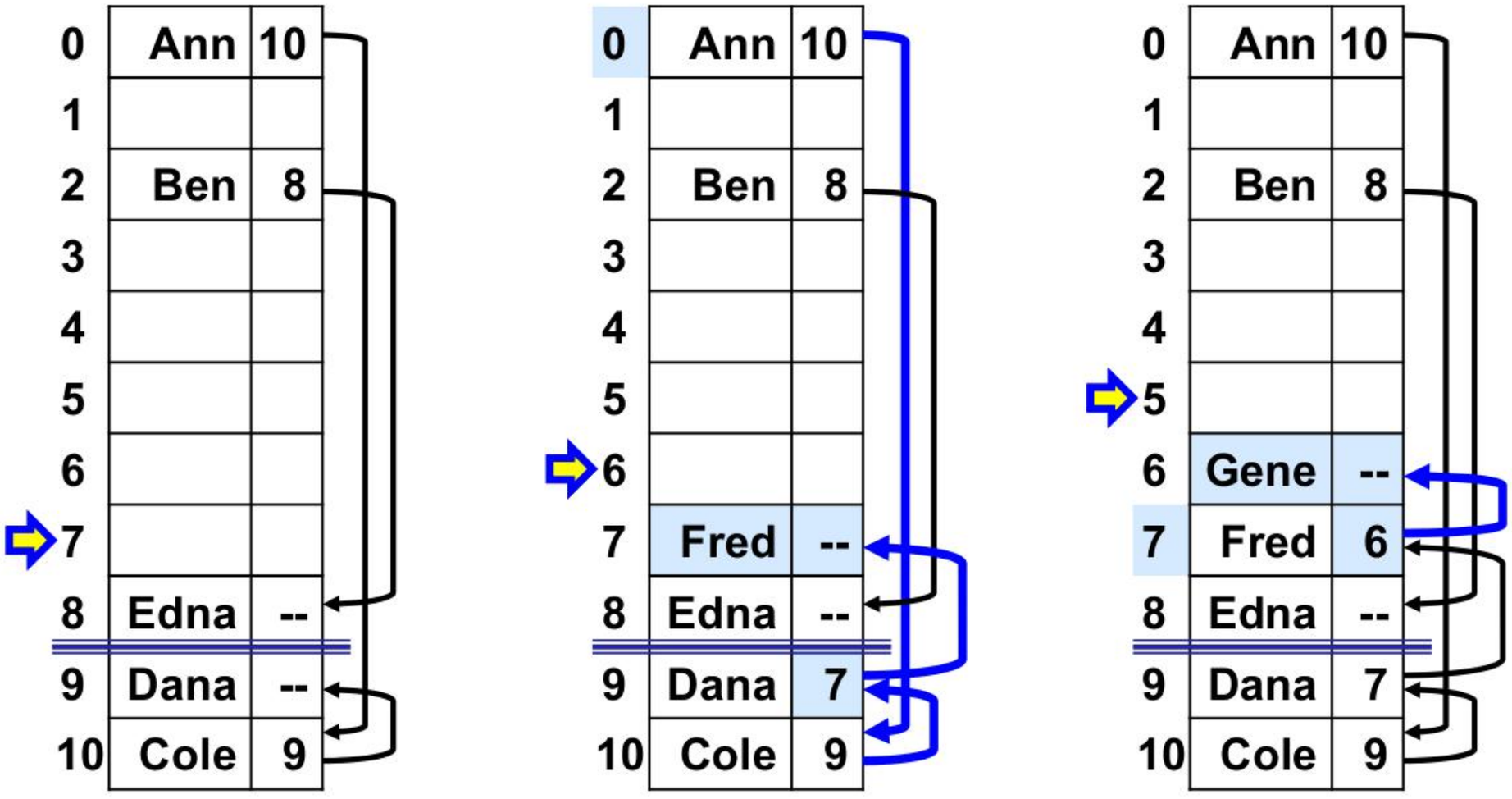






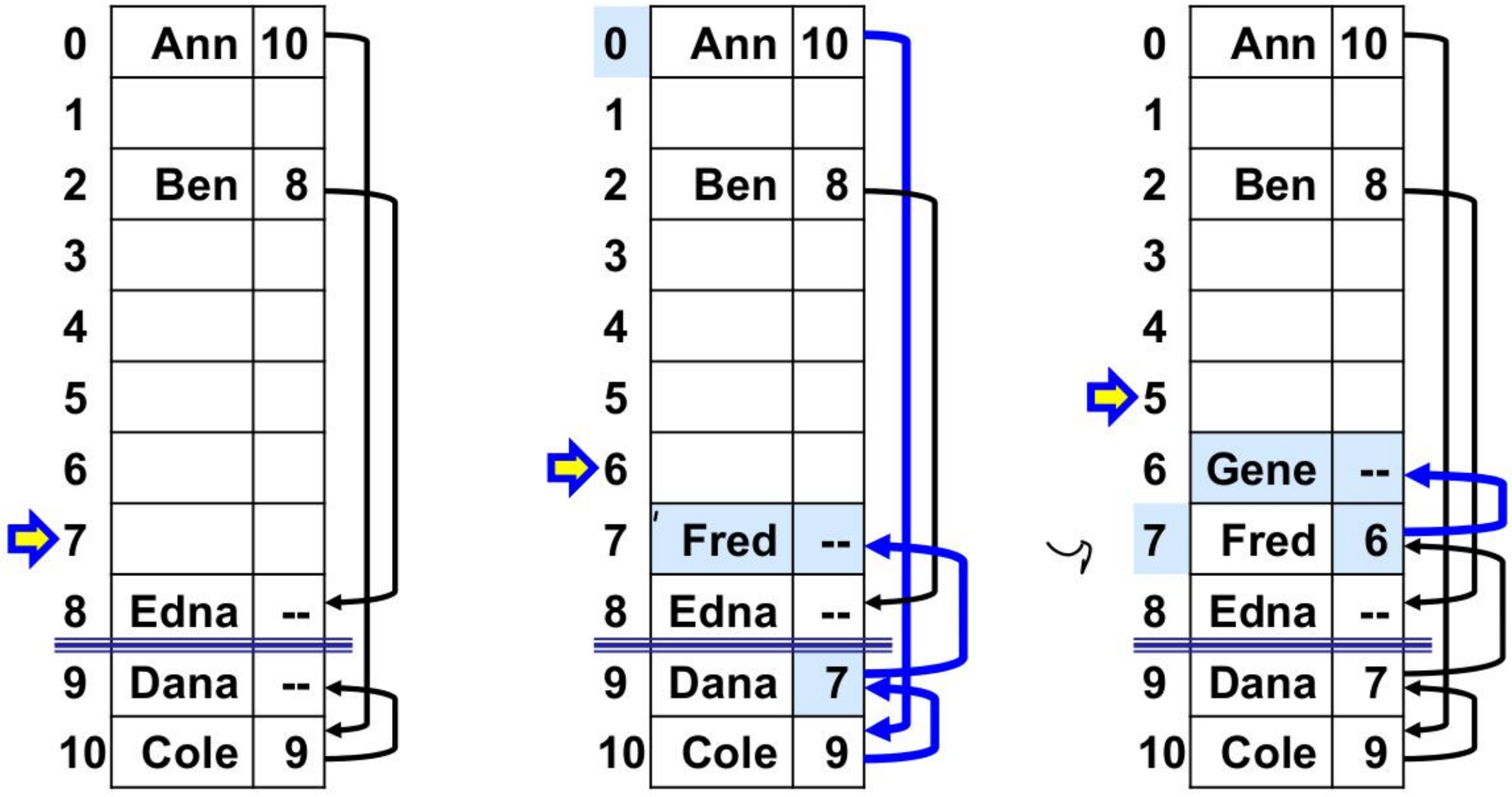
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)



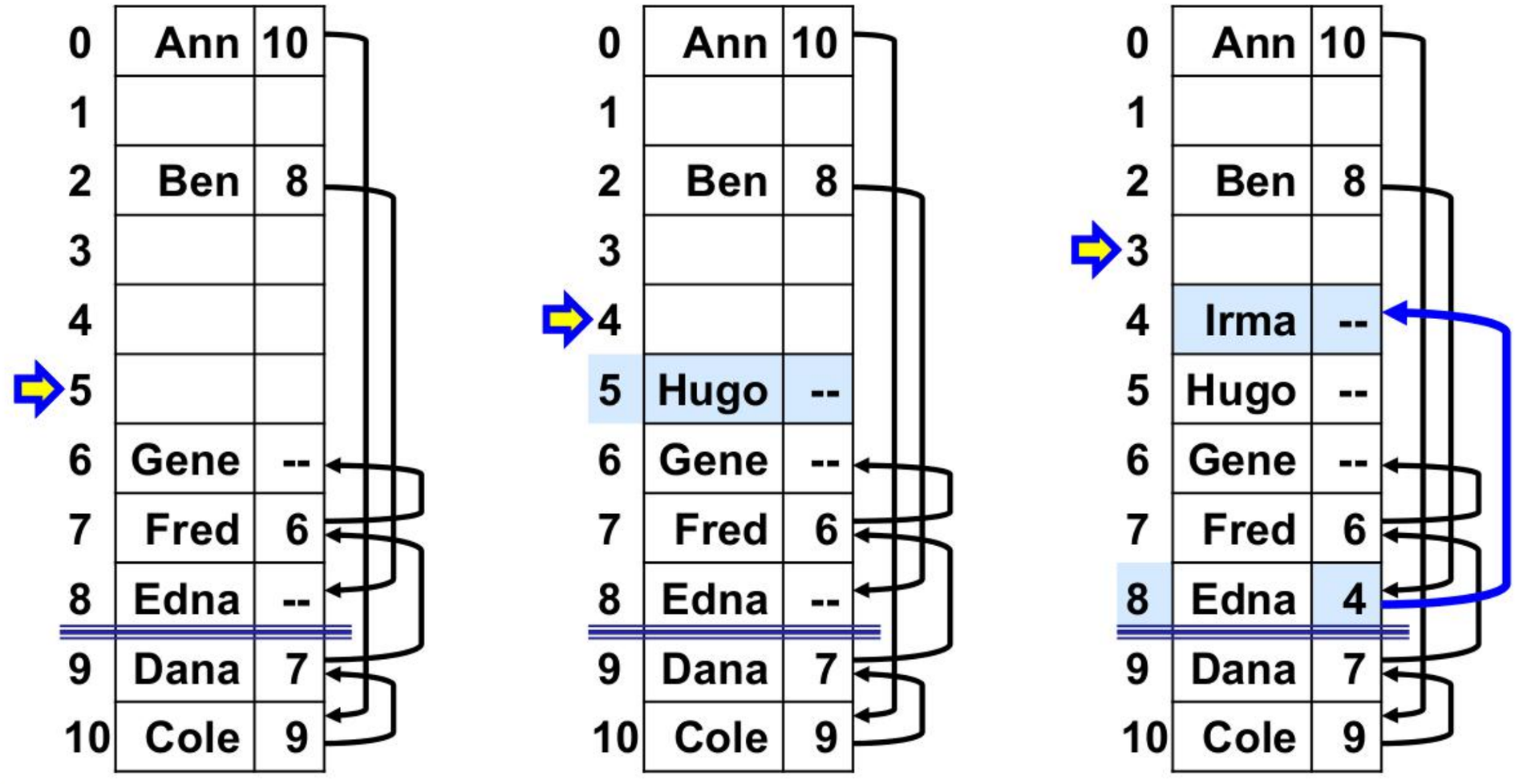
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)



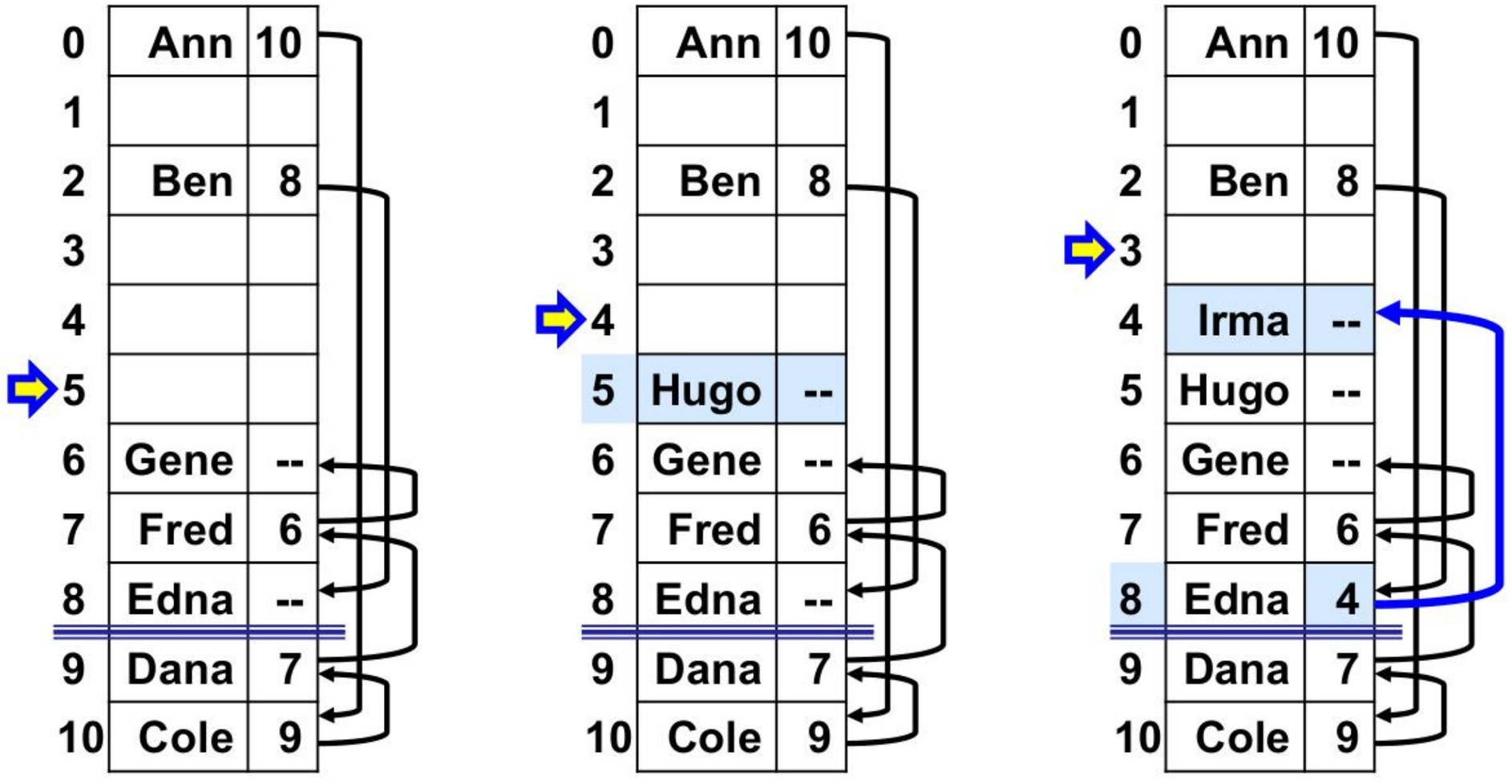
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)



data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)



data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

EICH (early insert coalesced hashing)

0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	10
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10	Cole	--

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

EICH (early insert coalesced hashing)

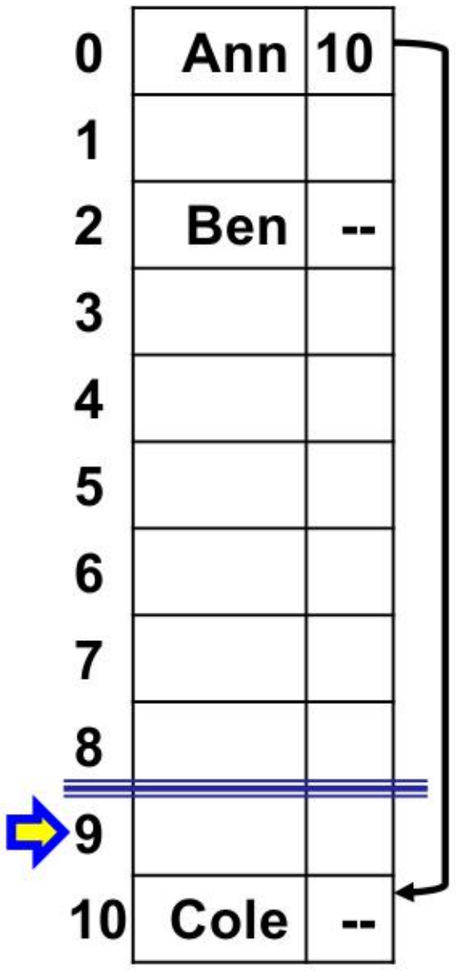
0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

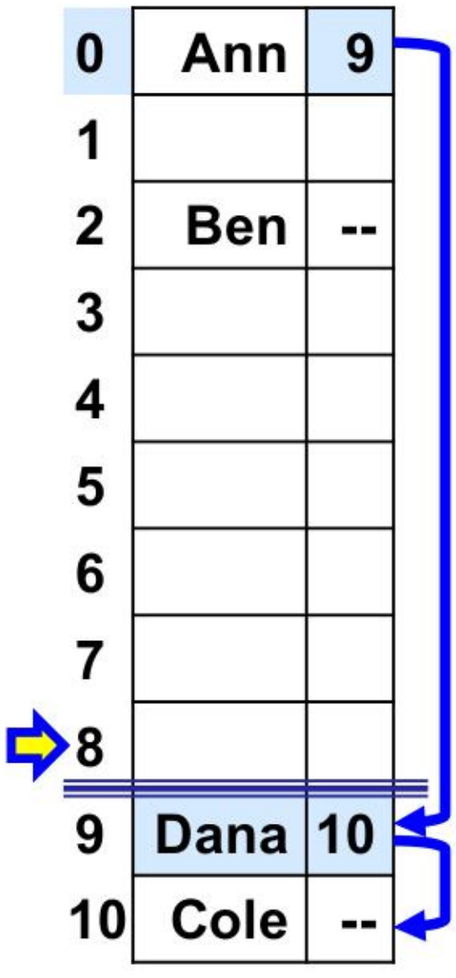
0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10		

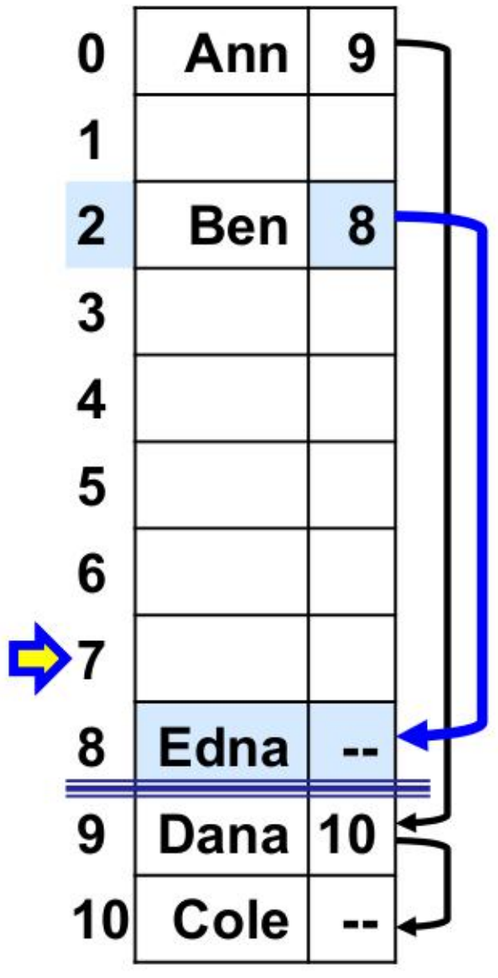
0	Ann	10
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10	Cole	--

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

EICH (early insert coalesced hashing)

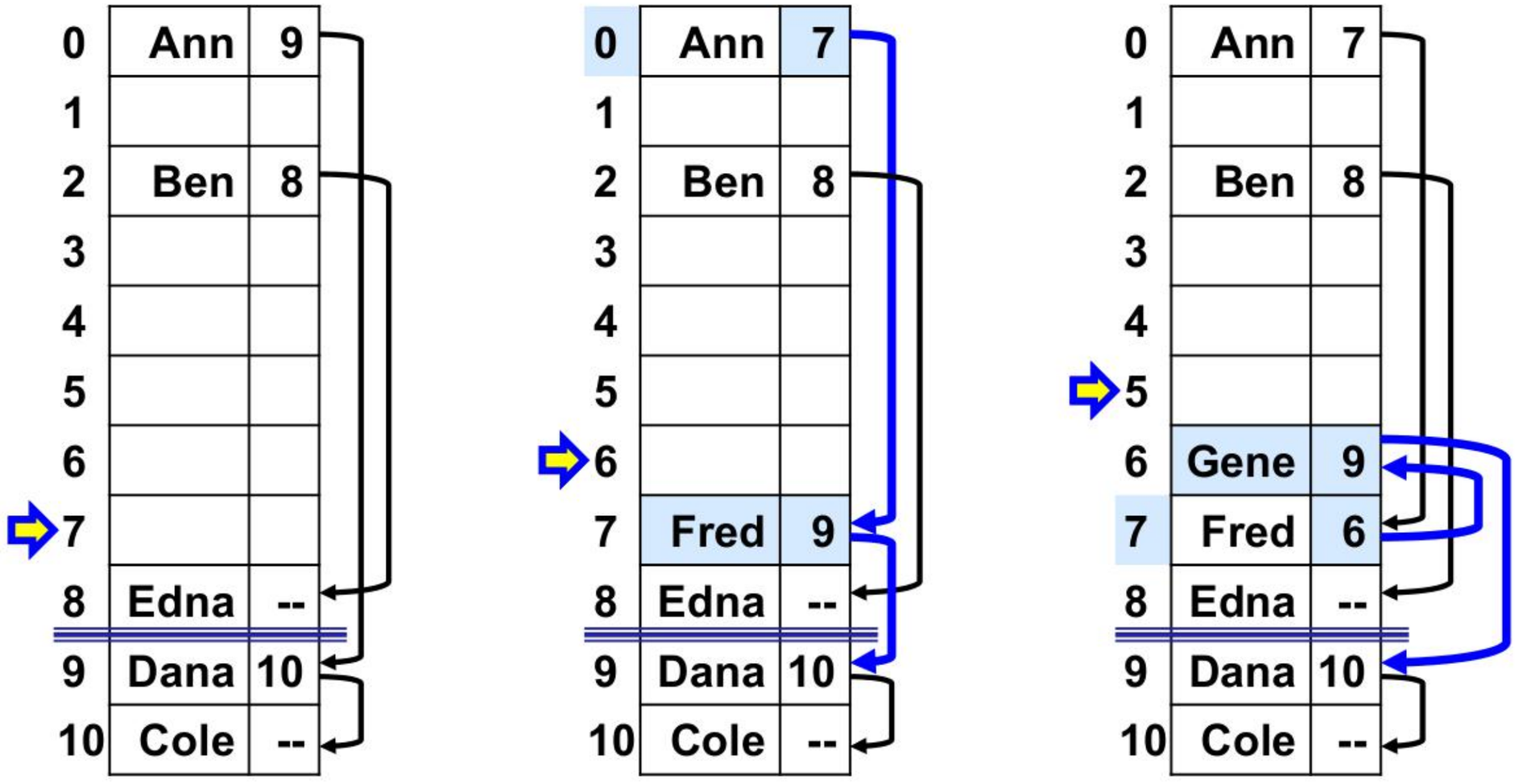






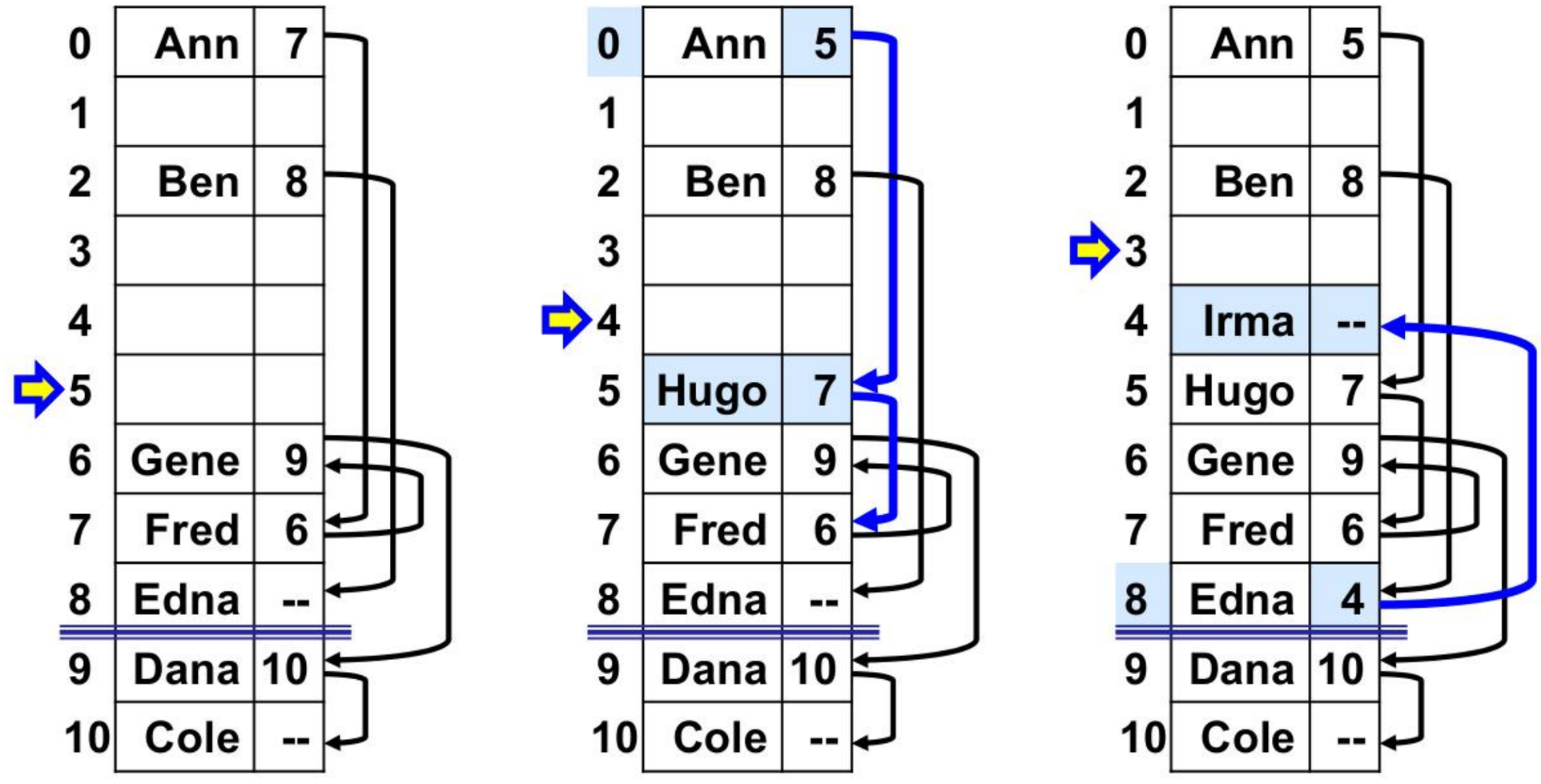
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

EICH (early insert coalesced hashing)



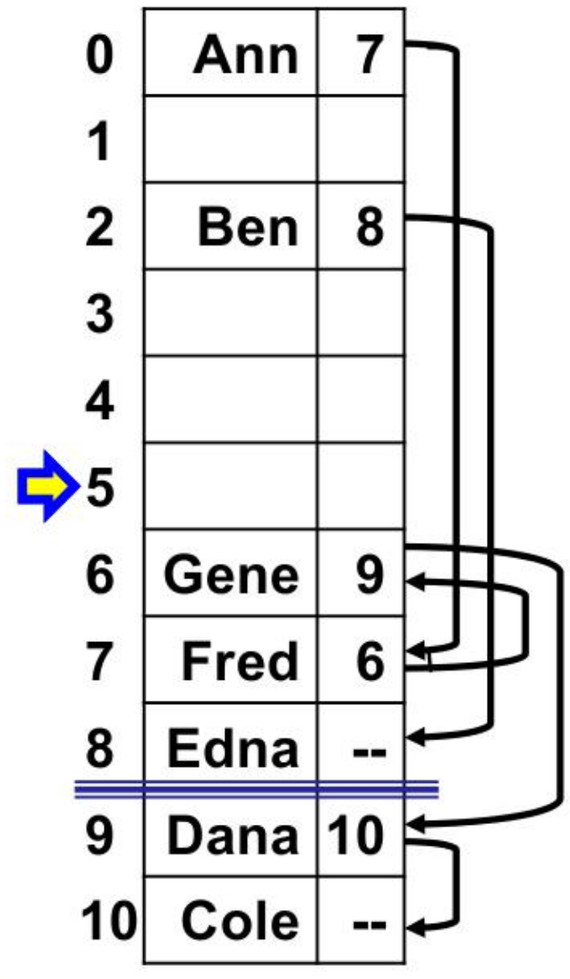
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

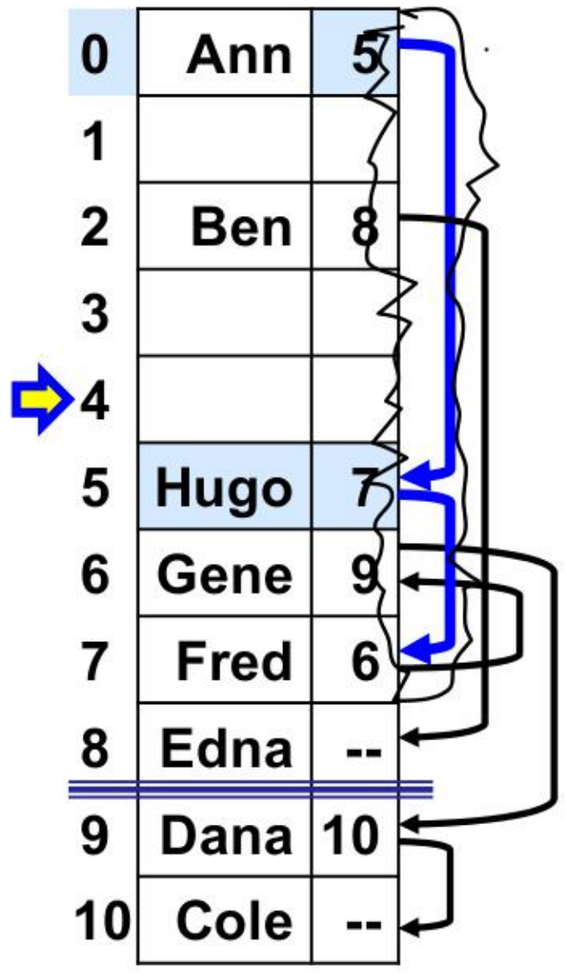
EICH (early insert coalesced hashing)

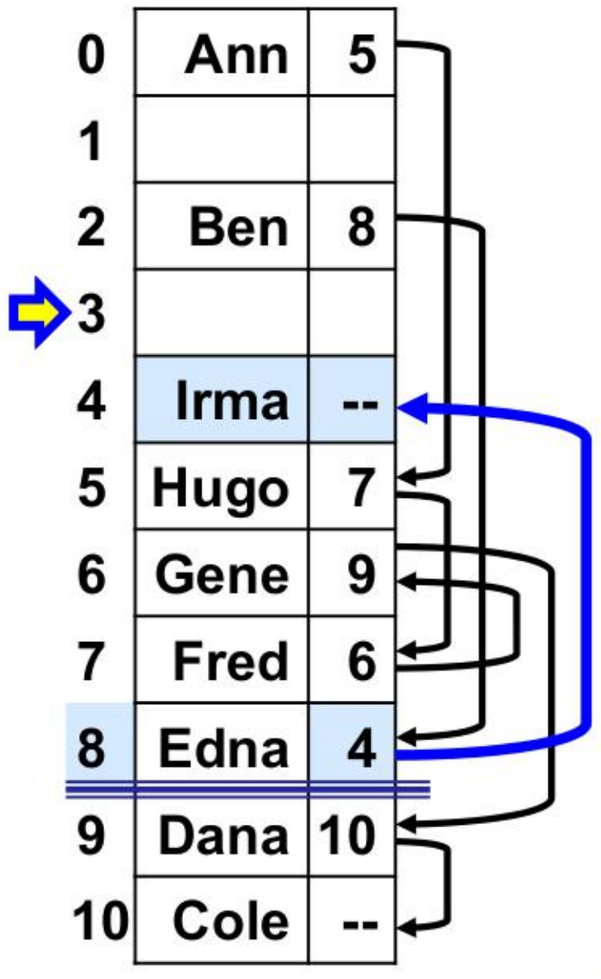


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

EICH (early insert coalesced hashing)







data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

VICH (variable insert coalesced hashing)

0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	10
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10	Cole	--

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	2	0	6

VICH (variable insert coalesced hashing)

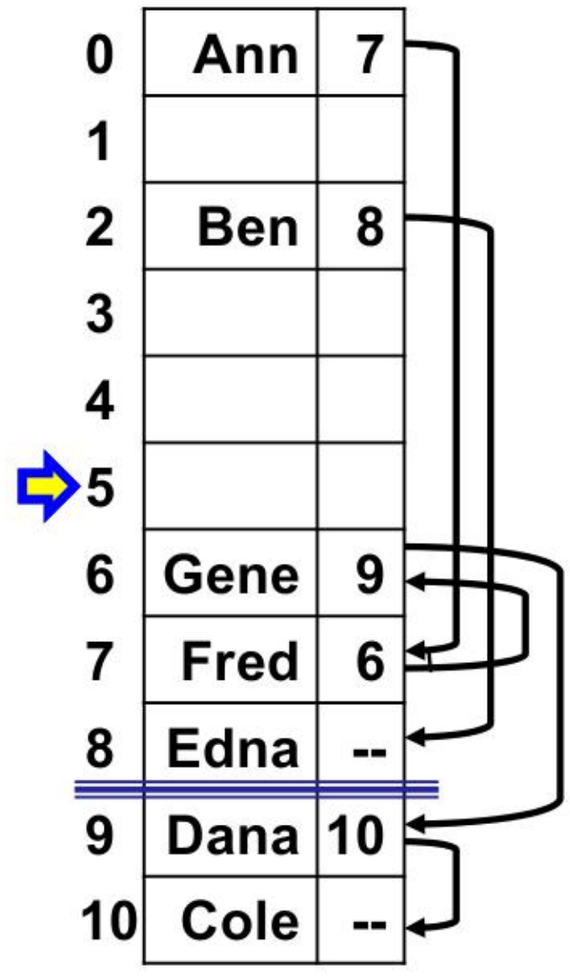
0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

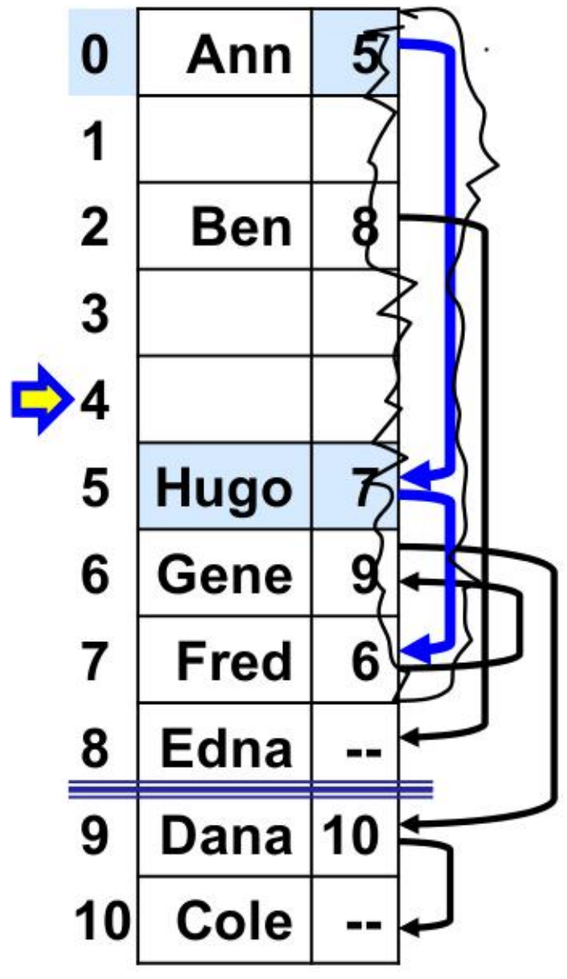
0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10		

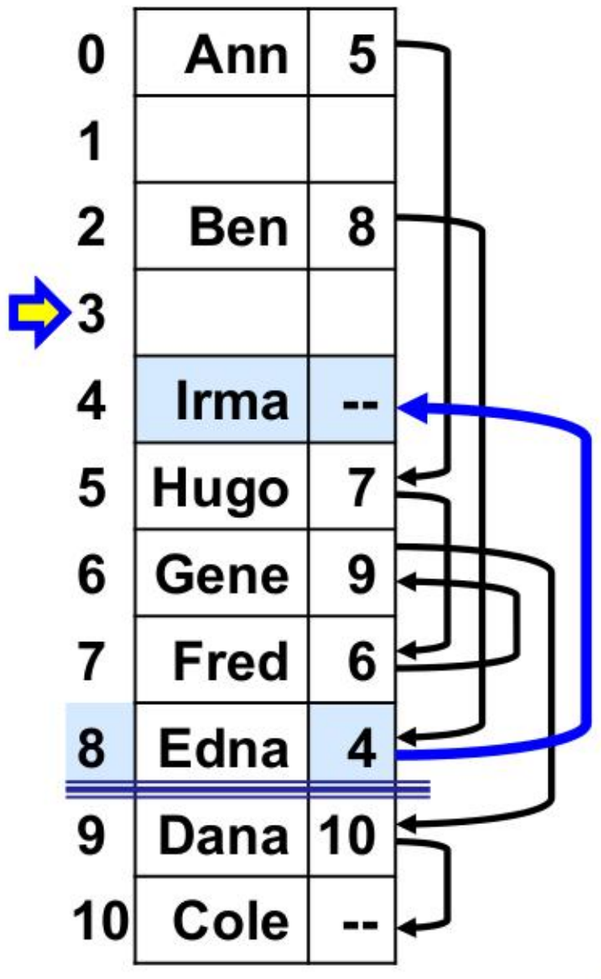
0	Ann	10
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10	Cole	--

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	2	0	6

EICH (early insert coalesced hashing)

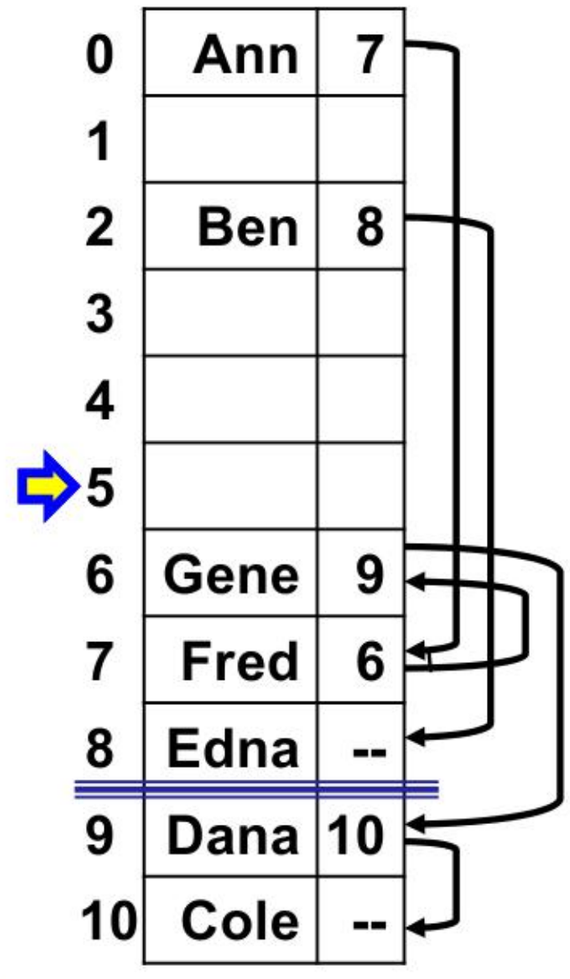


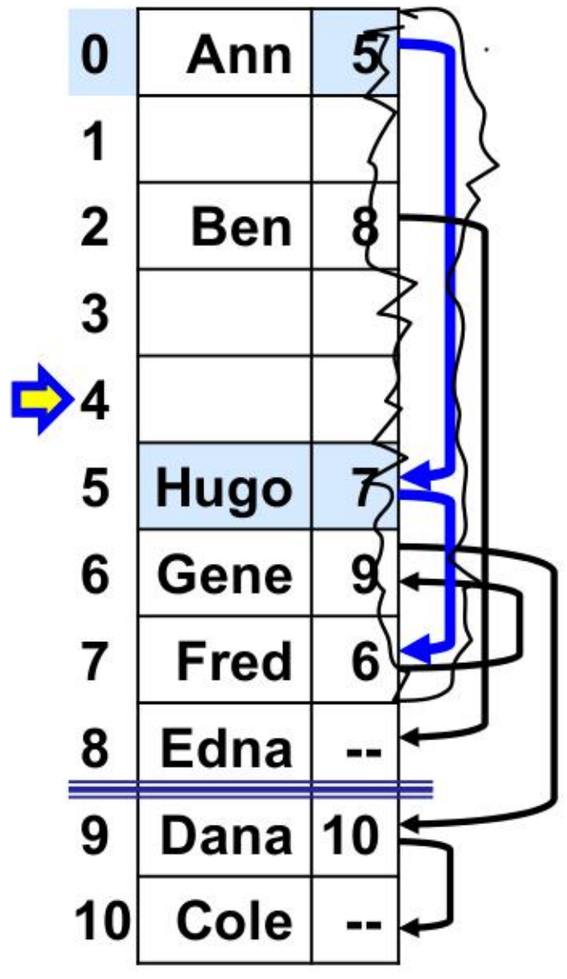


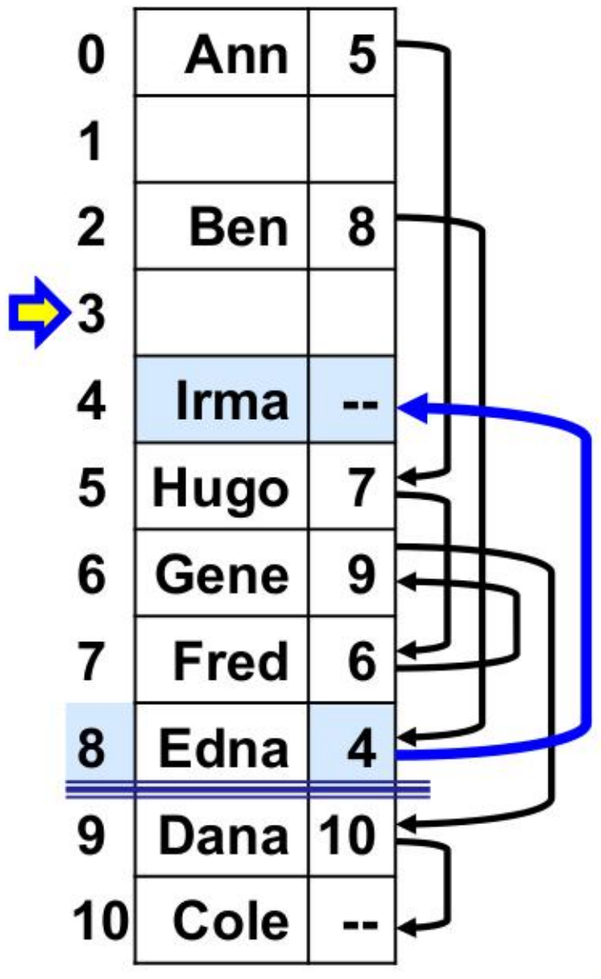


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

EICH (early insert coalesced hashing)

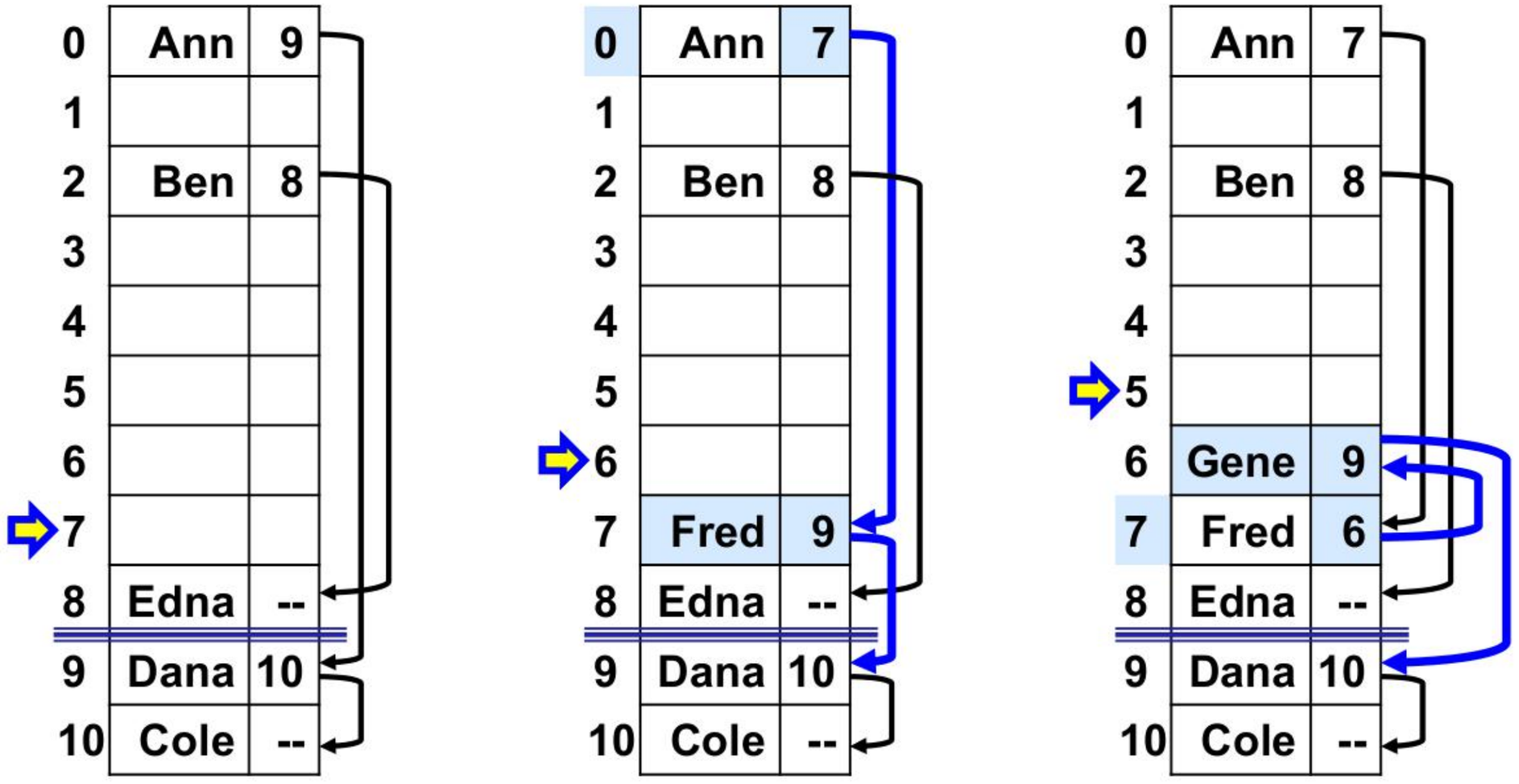






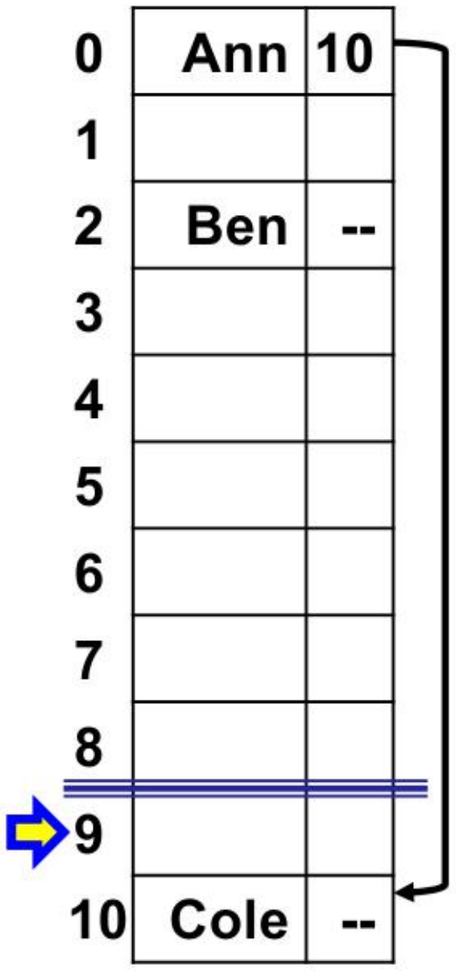
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

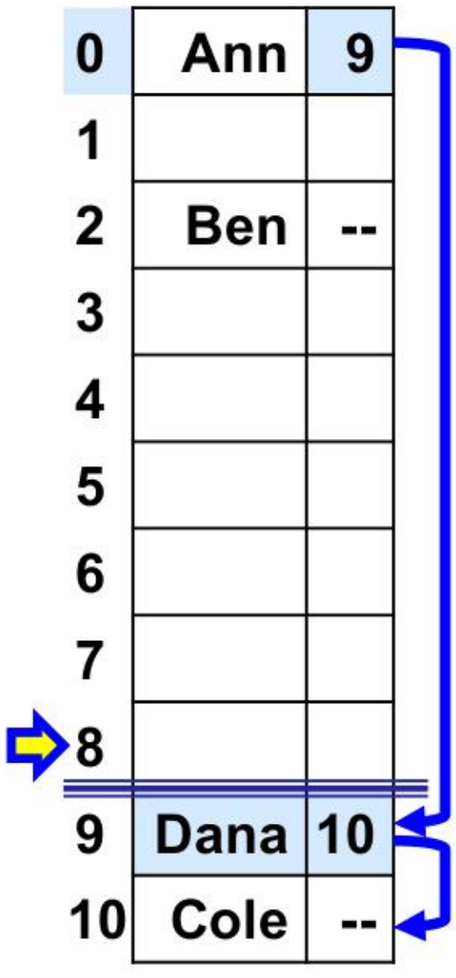
EICH (early insert coalesced hashing)

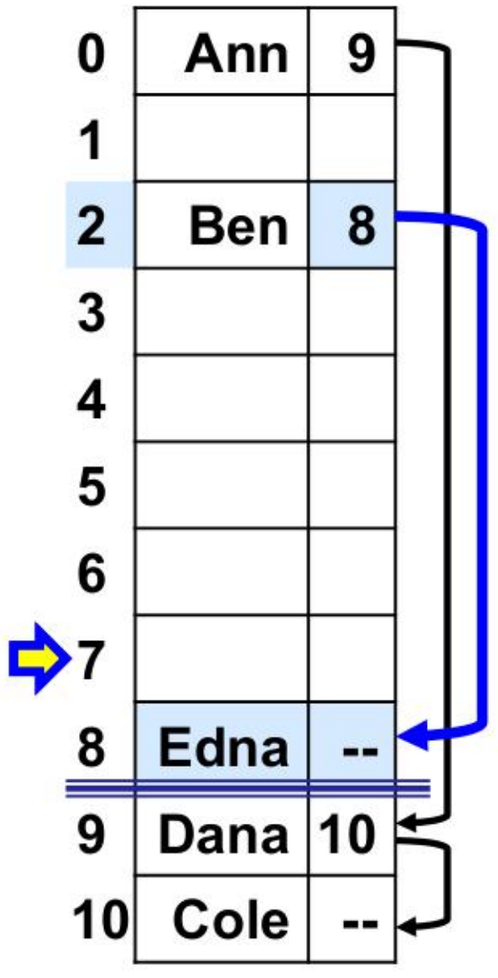


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

EICH (early insert coalesced hashing)







data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

EICH (early insert coalesced hashing)

0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	10
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10	Cole	--

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

EICH (early insert coalesced hashing)

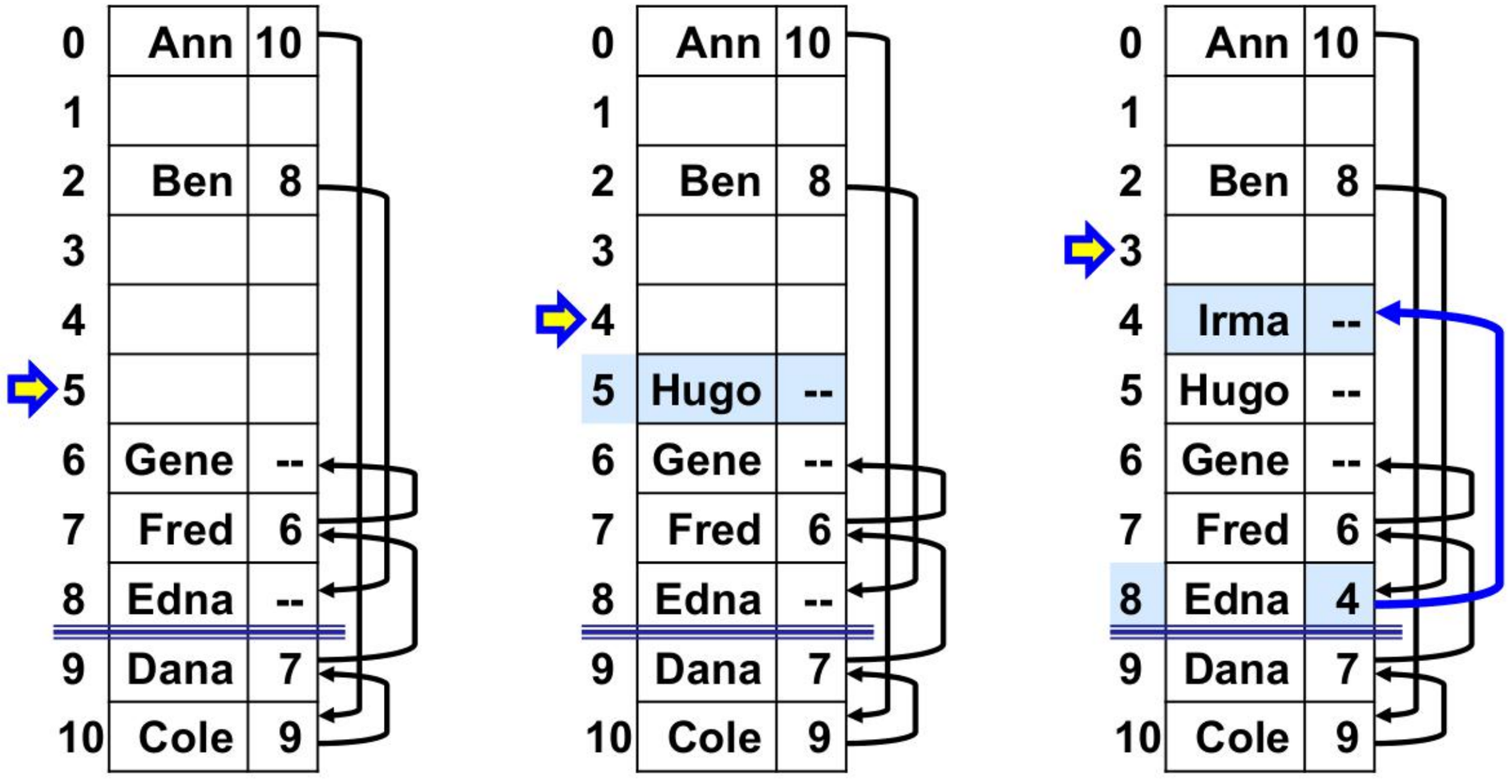
0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	10
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10	Cole	--

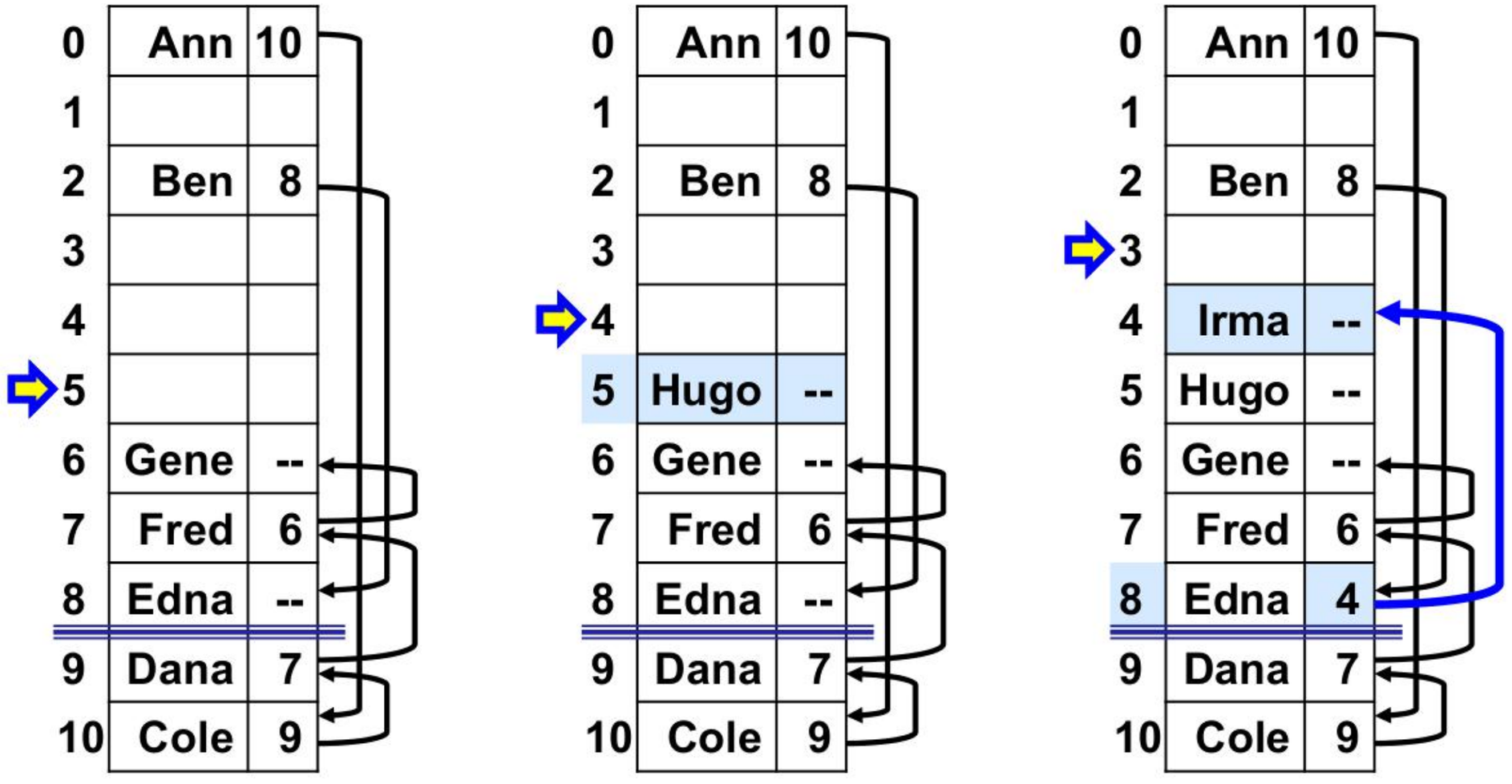
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

LICH (late insert coalesced hashing)



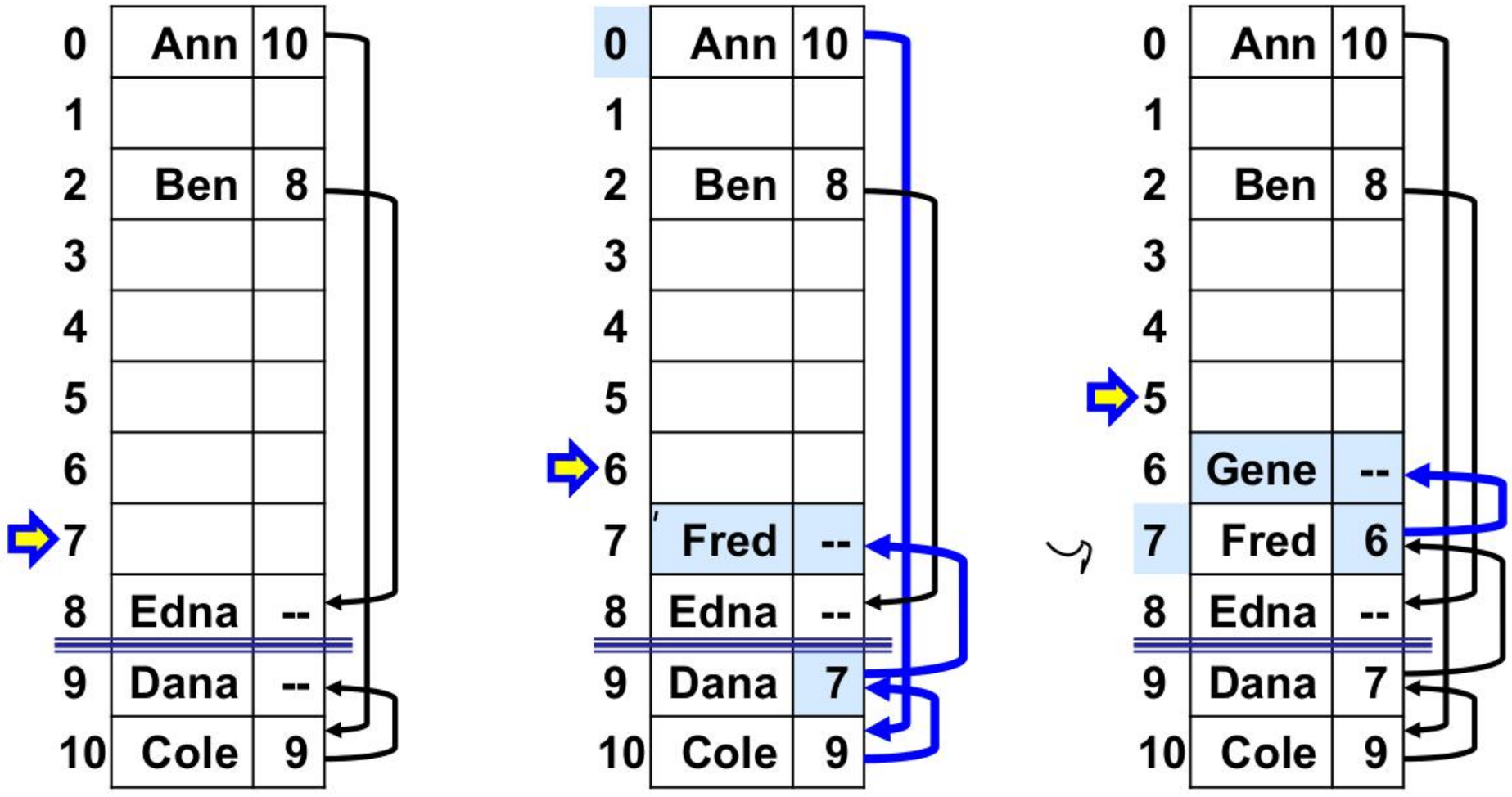
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)



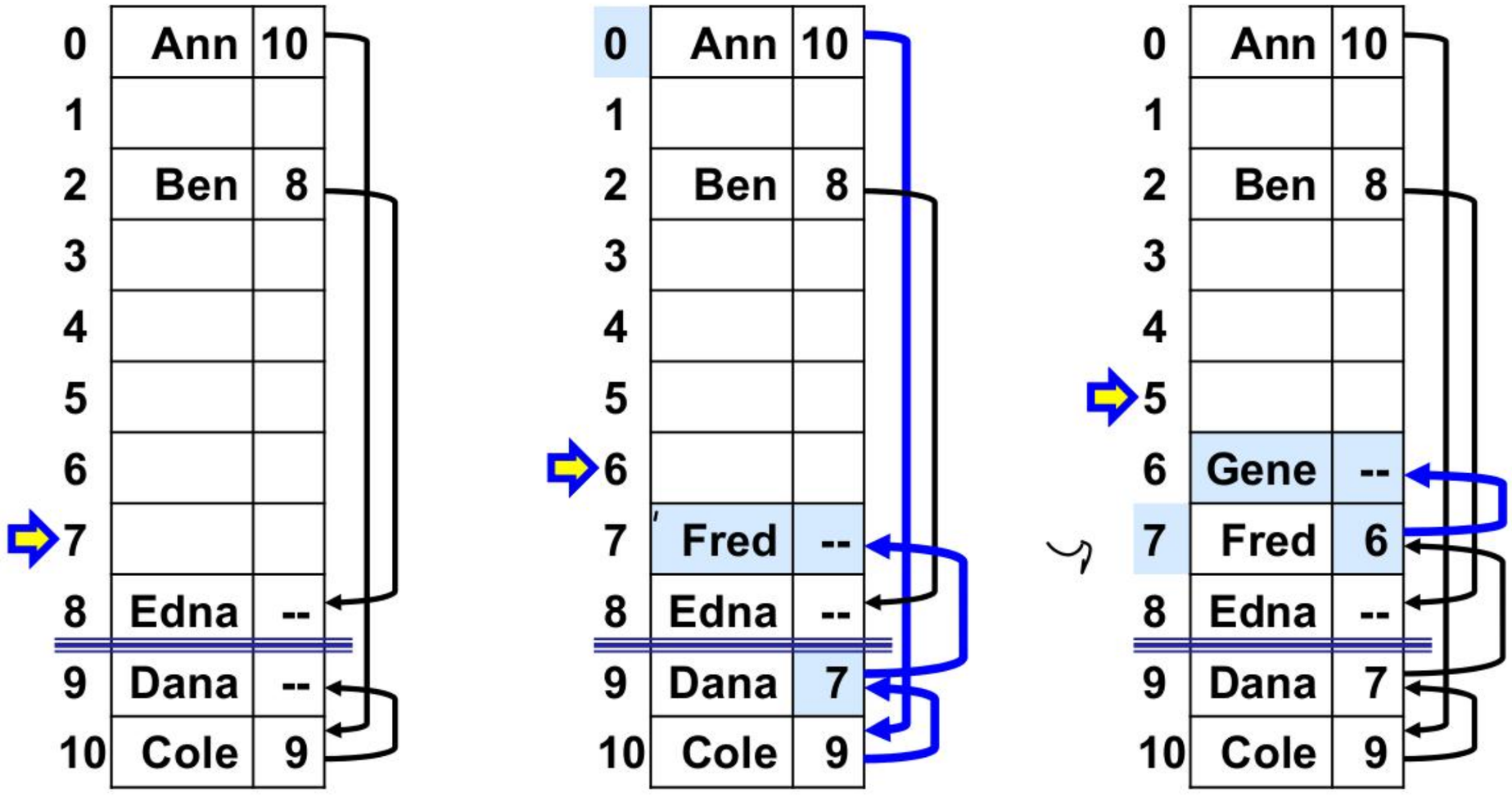
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)



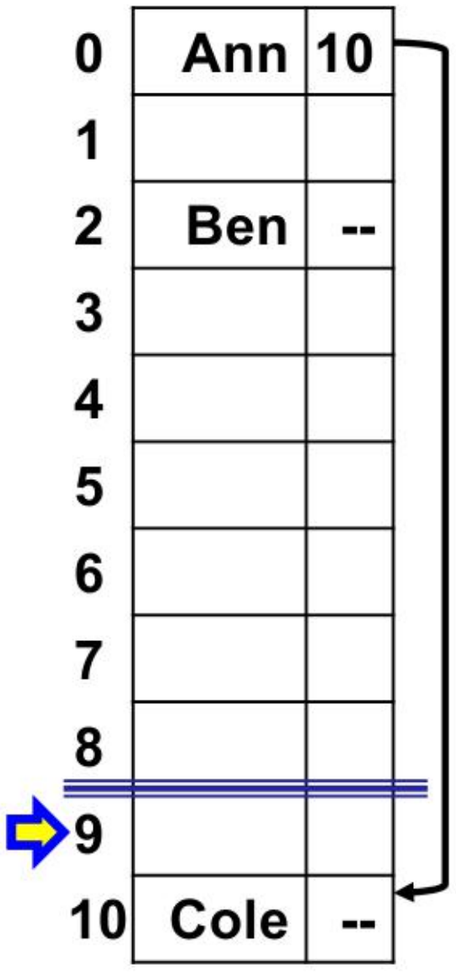
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

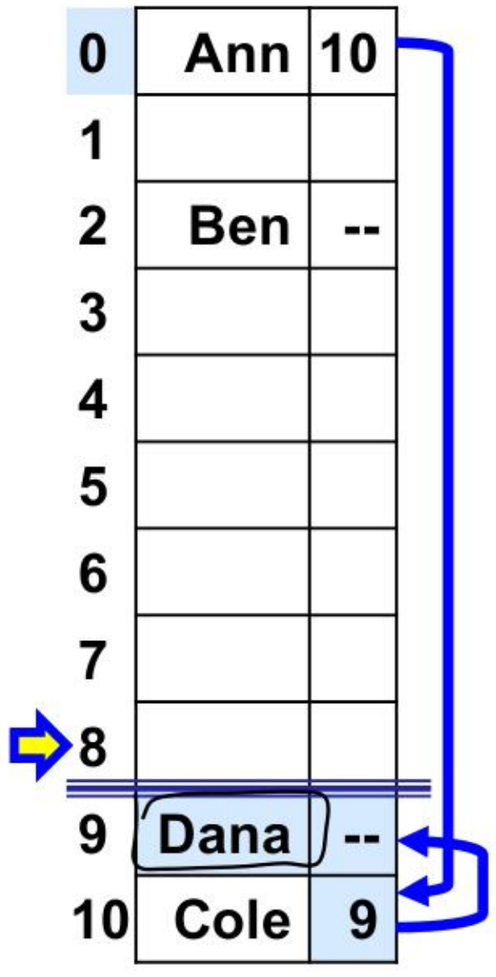
LICH (late insert coalesced hashing)

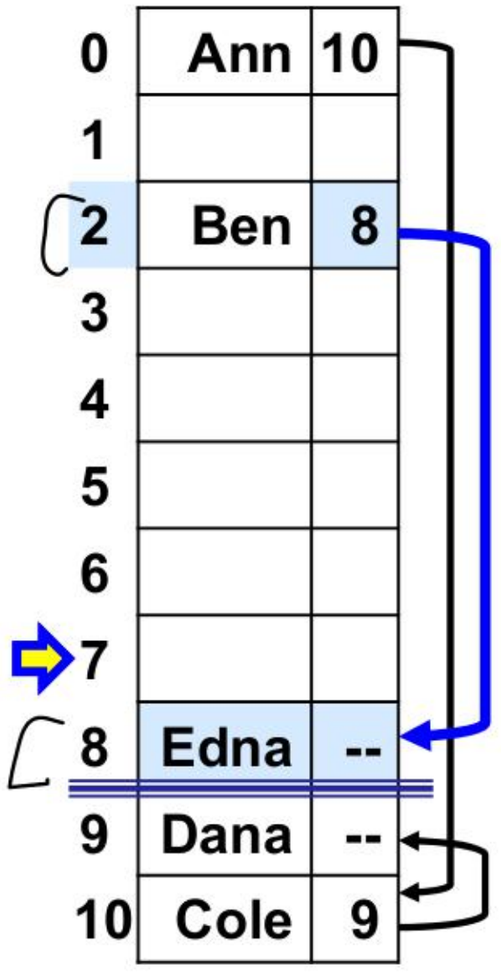


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)

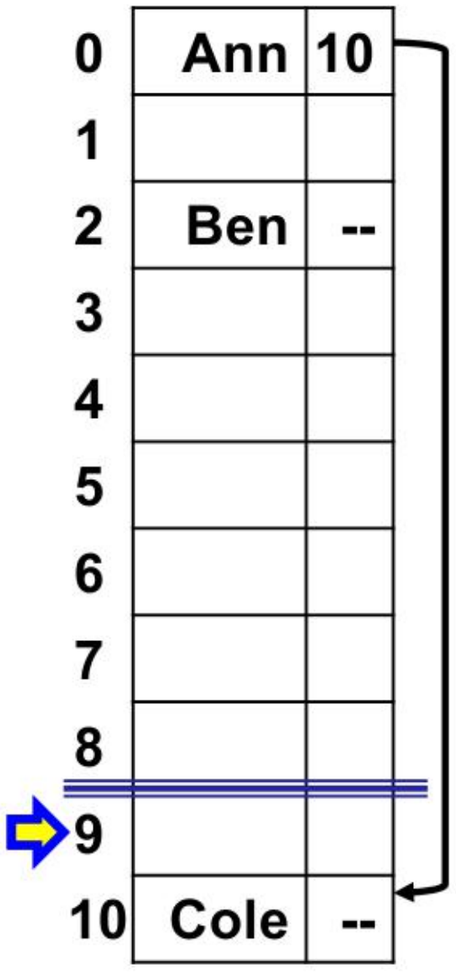


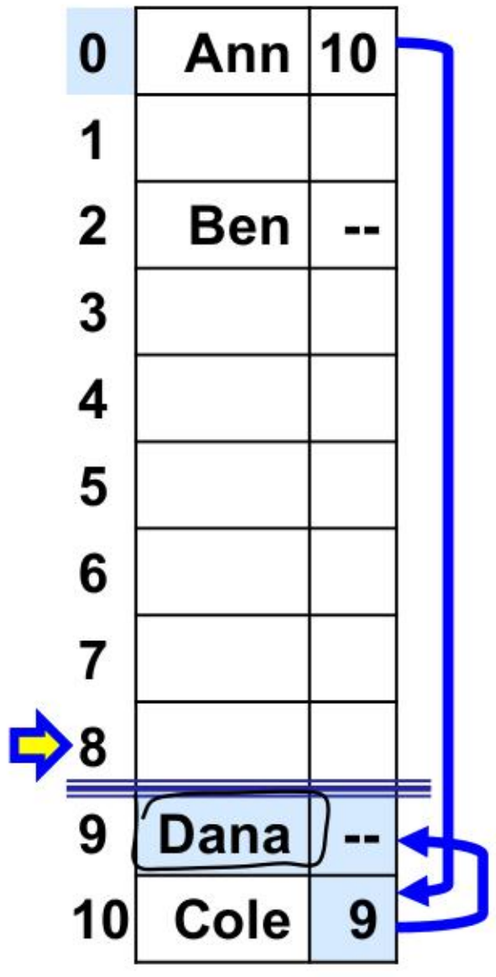


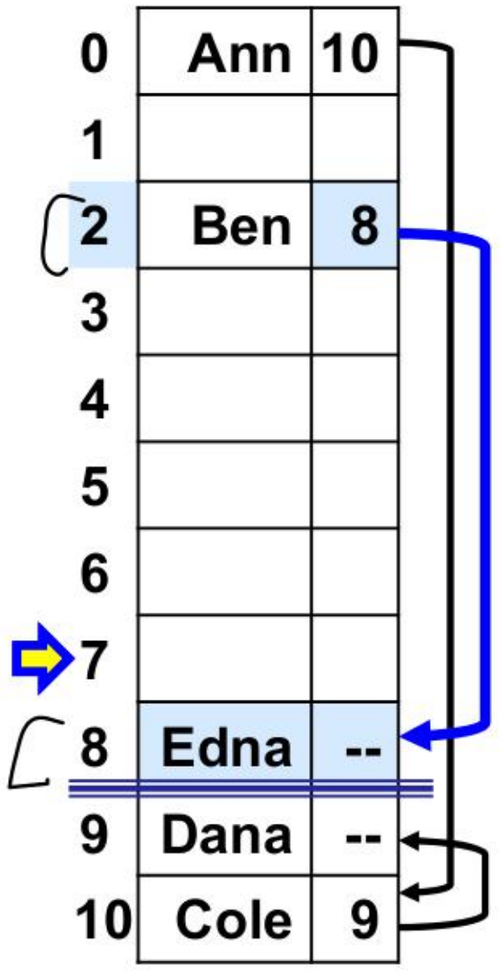


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)







data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)

0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	10
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10	Cole	--

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)

0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	10
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10	Cole	--

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

Srůstající hashování s pomocnou pamětí

Pro snížení srůstání a tedy zvýšení efektivity hashování se tabulka rozšiřuje o pomocnou paměť - tzv. sklep (cellar).

Sklep je místo na konci tabulky, které není adresovatelné hashovací funkcí, má ale stejnou strukturu jako celá tabulka.

Algoritmy LICH a EICH jsou analogické varianty algoritmů LISCH a EISCH s přidáním sklepa.
Po naplnění sklepa pokračuje plnění jako v LISCH a EISCH.

Algoritmus VICH (variable insert coalesced hashing) připojuje prvek za poslední prvek seznamu, který je ještě ve sklepě. Pokud ve sklepě žádný není, vkládá jako EISCH, tj. hned za kolidující prvek v seznamu.

Srůstající hashování s pomocnou pamětí

Pro snížení srůstání a tedy zvýšení efektivity hashování se tabulka rozšiřuje o pomocnou paměť - tzv. sklep (cellar).

Sklep je místo na konci tabulky, které není adresovatelné hashovací funkcí, má ale stejnou strukturu jako celá tabulka.

Algoritmy LICH a EICH jsou analogické varianty algoritmů LISCH a EISCH s přidáním sklepa.
Po naplnění sklepa pokračuje plnění jako v LISCH a EISCH.

Algoritmus VICH (variable insert coalesced hashing) připojuje prvek za poslední prvek seznamu, který je ještě ve sklepě. Pokud ve sklepě žádný není, vkládá jako EISCH, tj. hned za kolidující prvek v seznamu.

LICH (late insert coalesced hashing)

0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	10
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10	Cole	--

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)

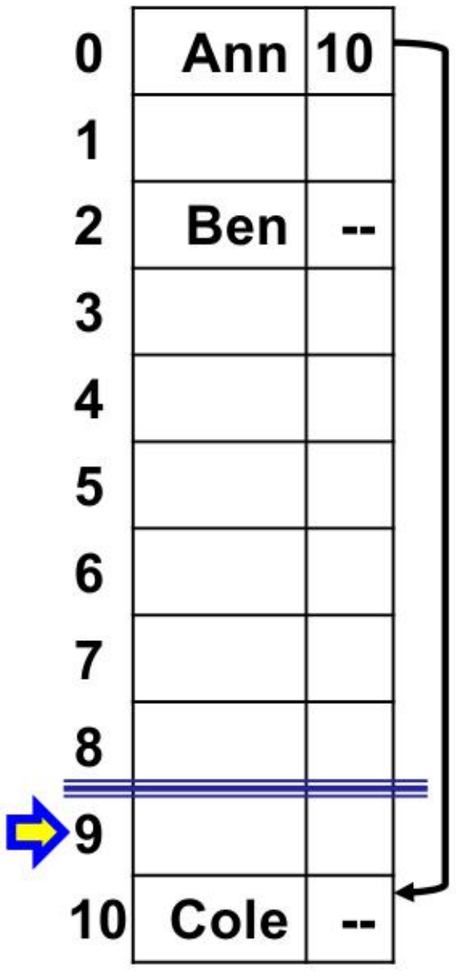
0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

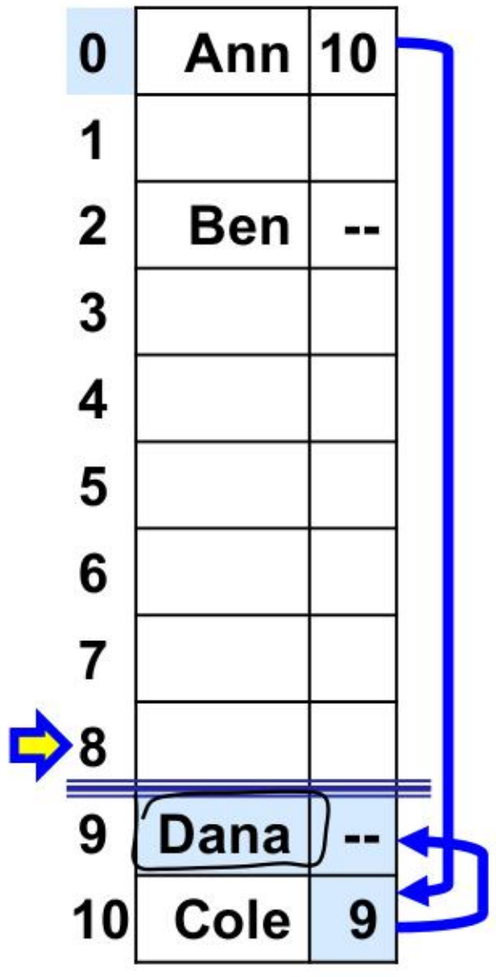
0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10		

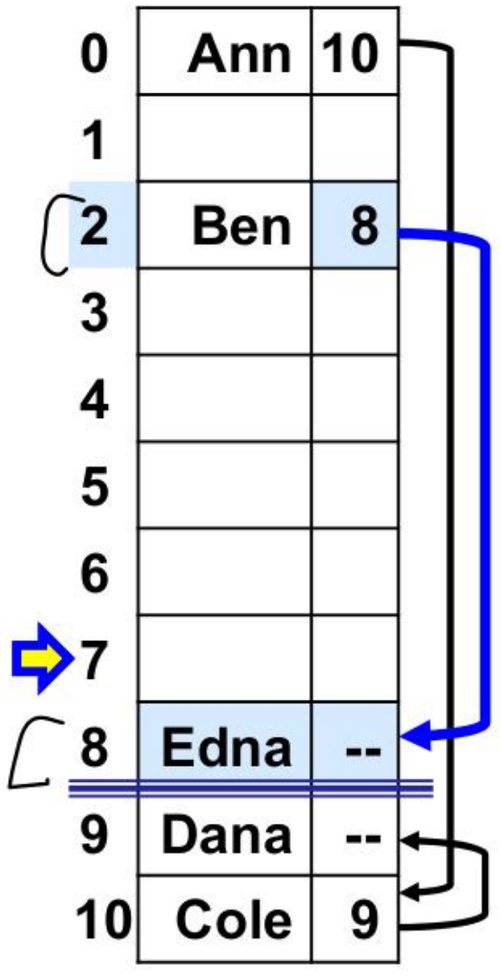
0	Ann	10
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10	Cole	--

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)

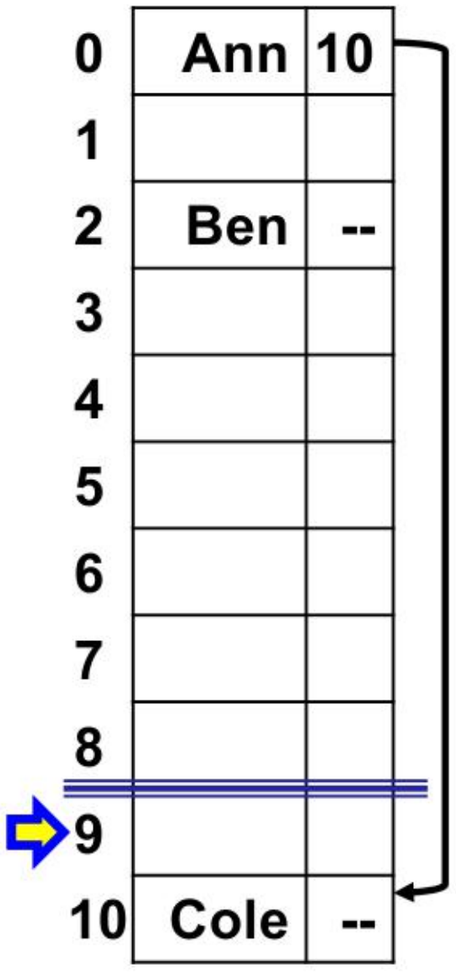


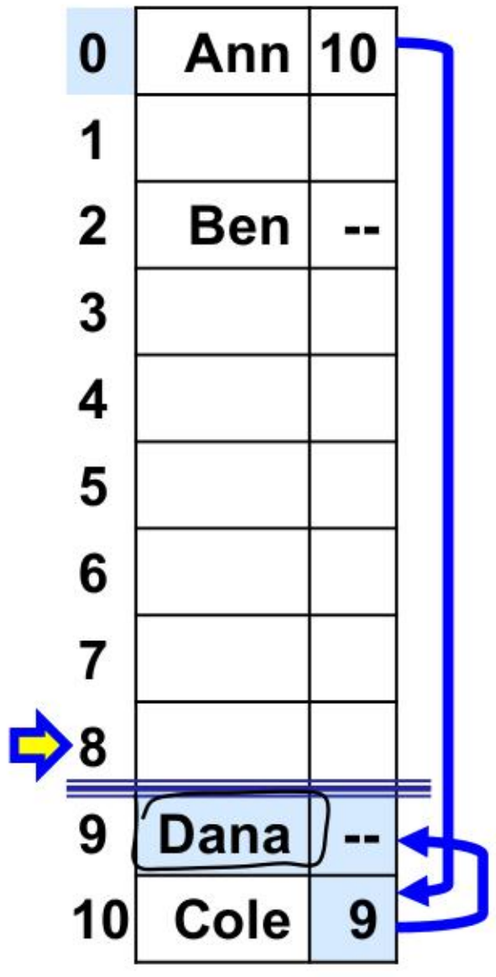


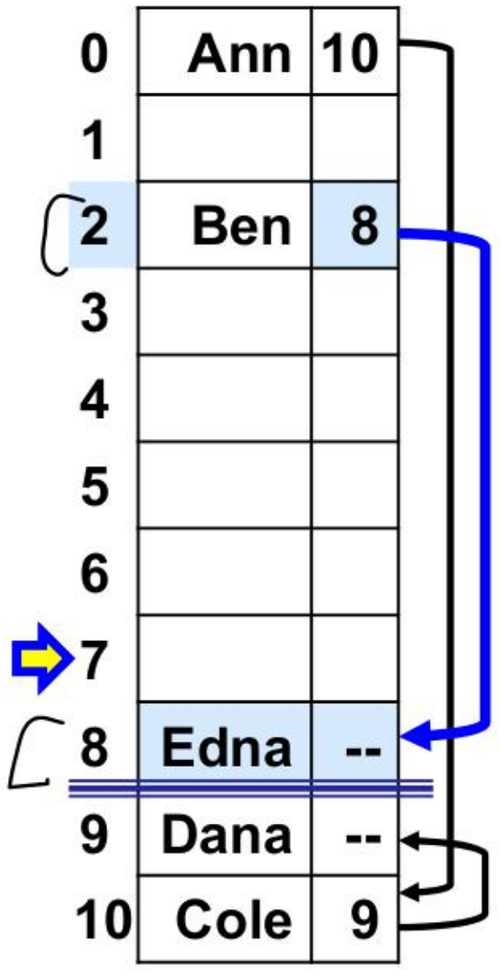


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)

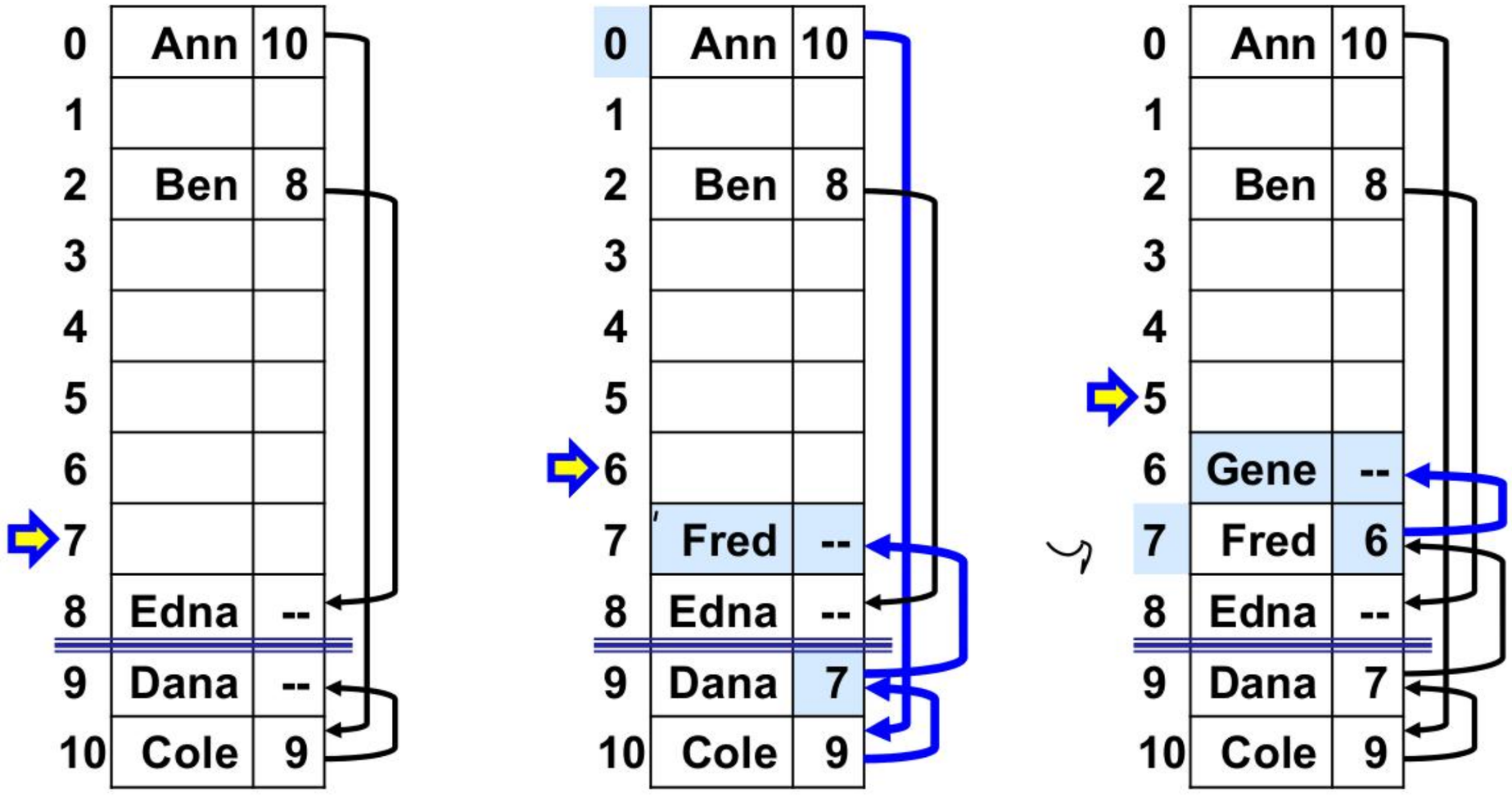






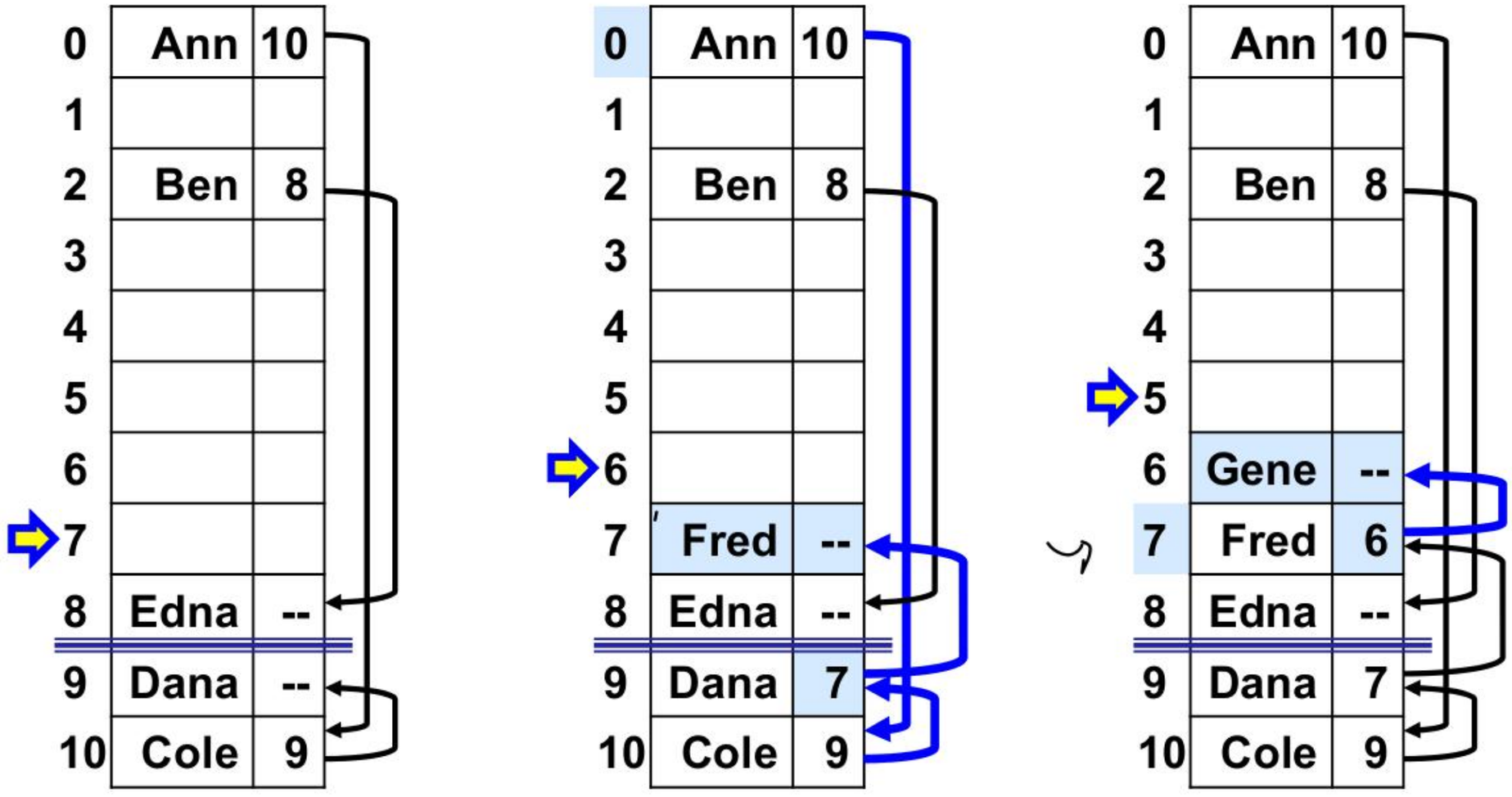
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)



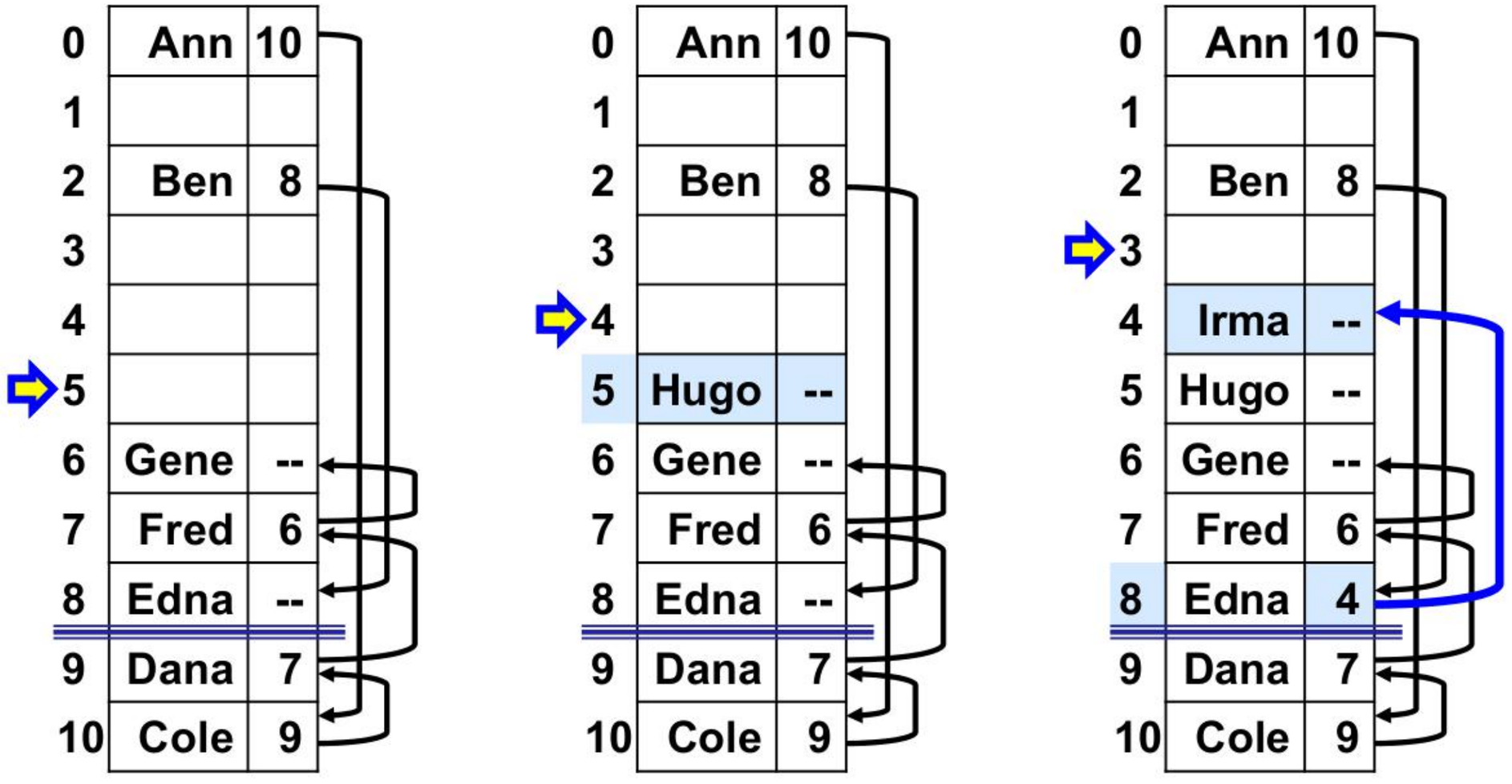
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)



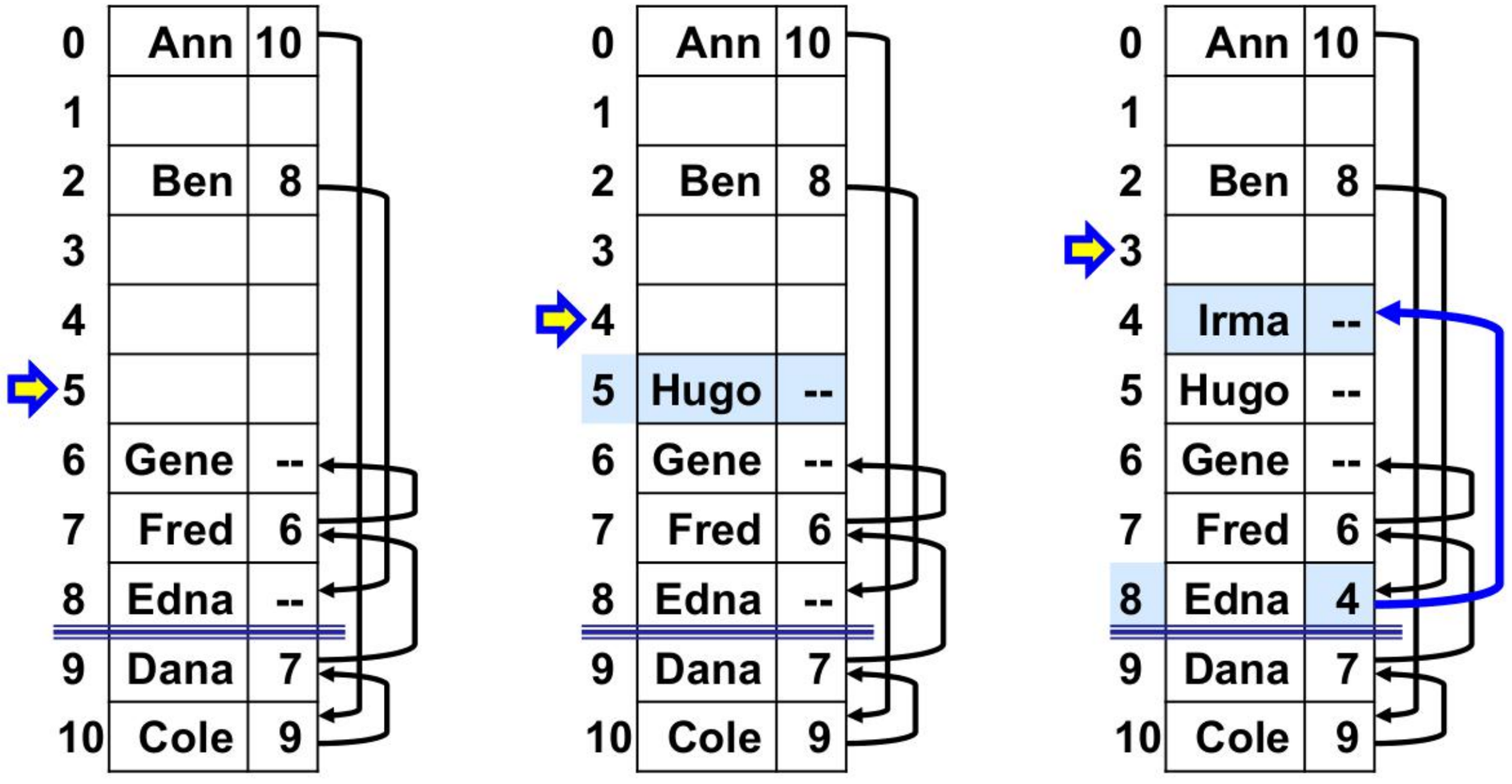
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)



data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

LICH (late insert coalesced hashing)



data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	5	8

EICH (early insert coalesced hashing)

0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	10
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10	Cole	--

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

EICH (early insert coalesced hashing)

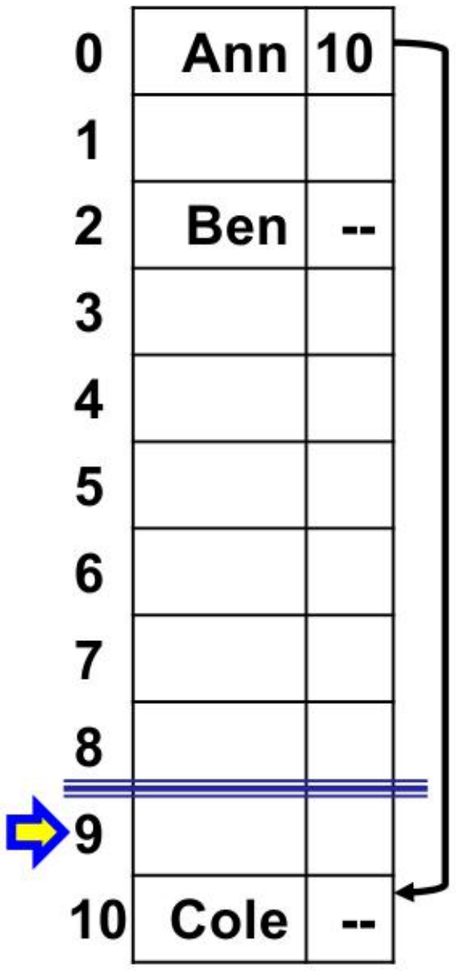
0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

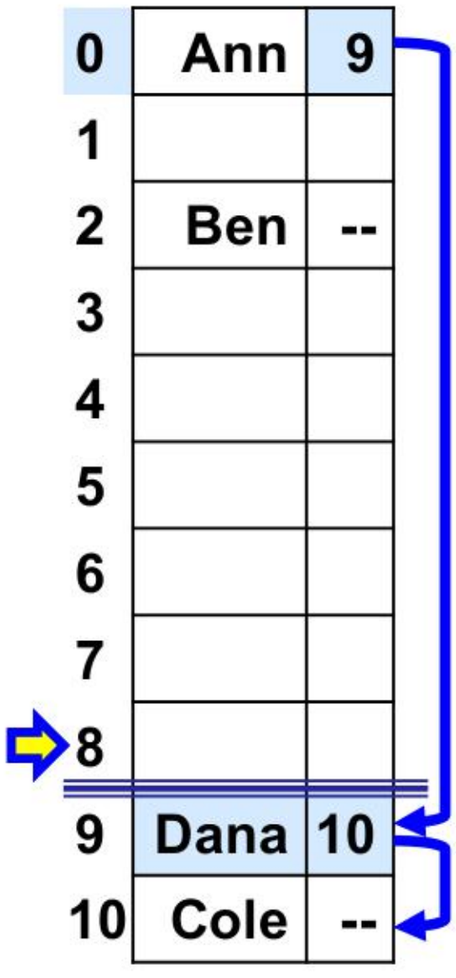
0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10		

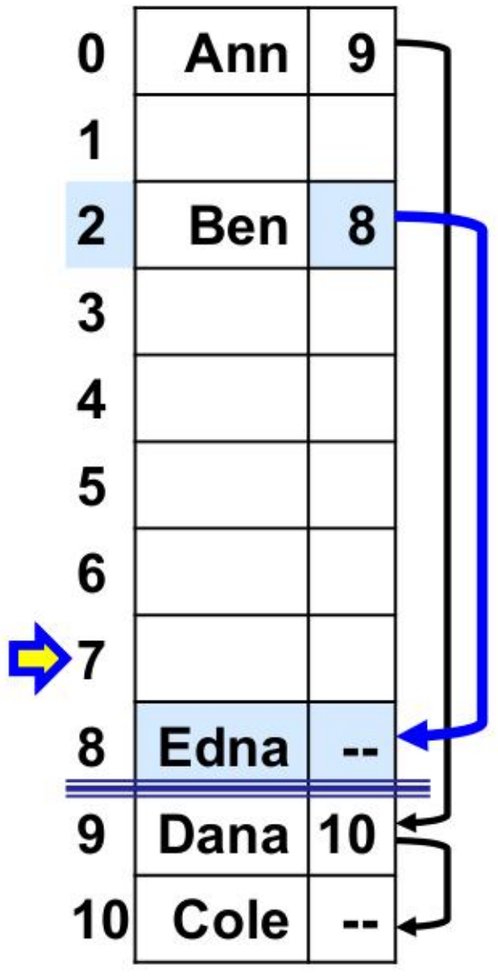
0	Ann	10
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10	Cole	--

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

EICH (early insert coalesced hashing)

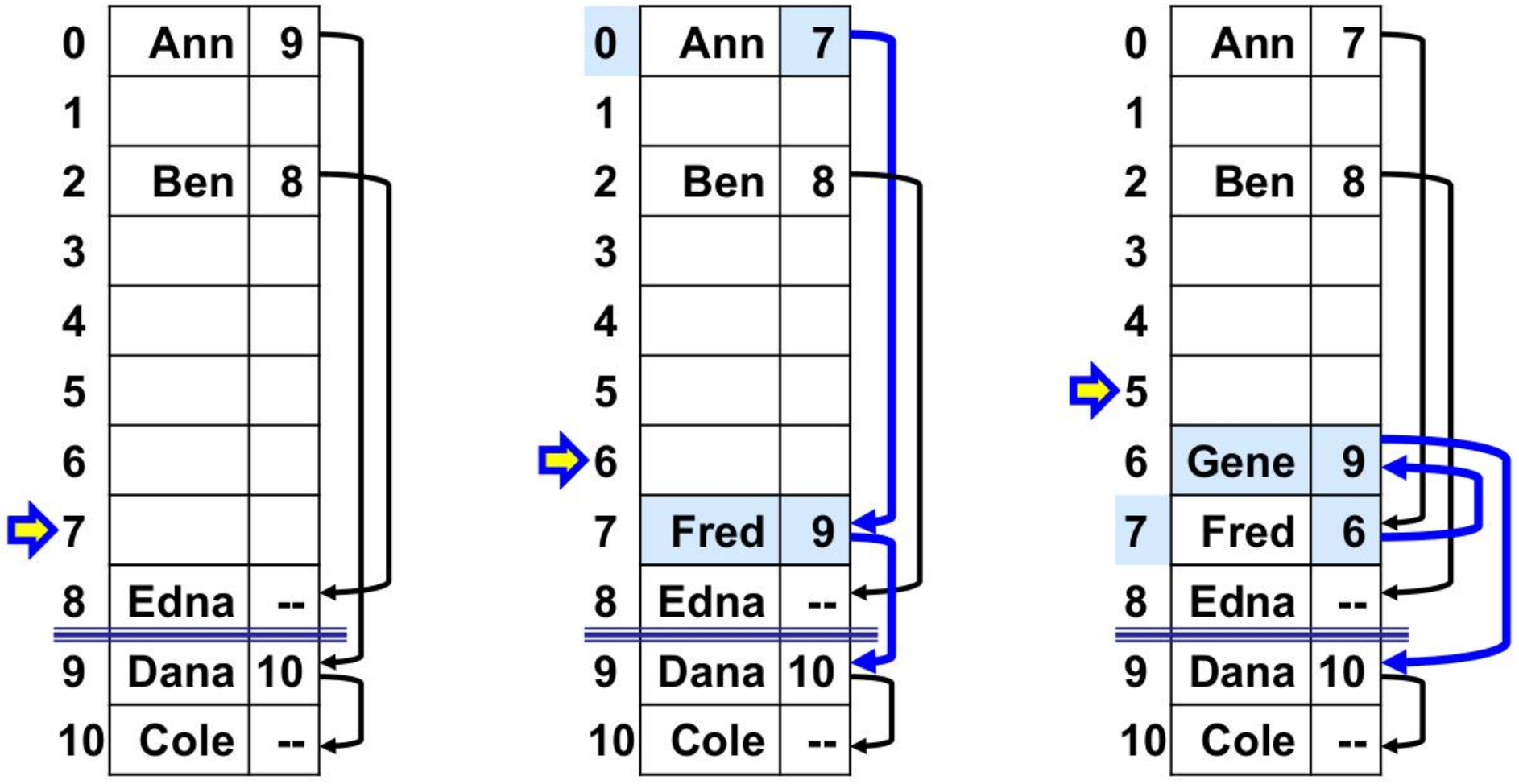






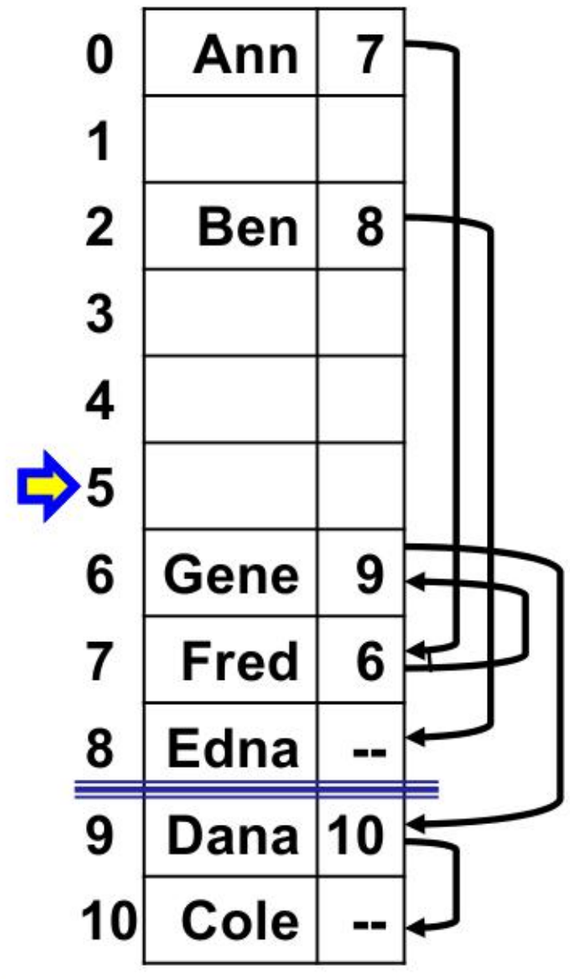
data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

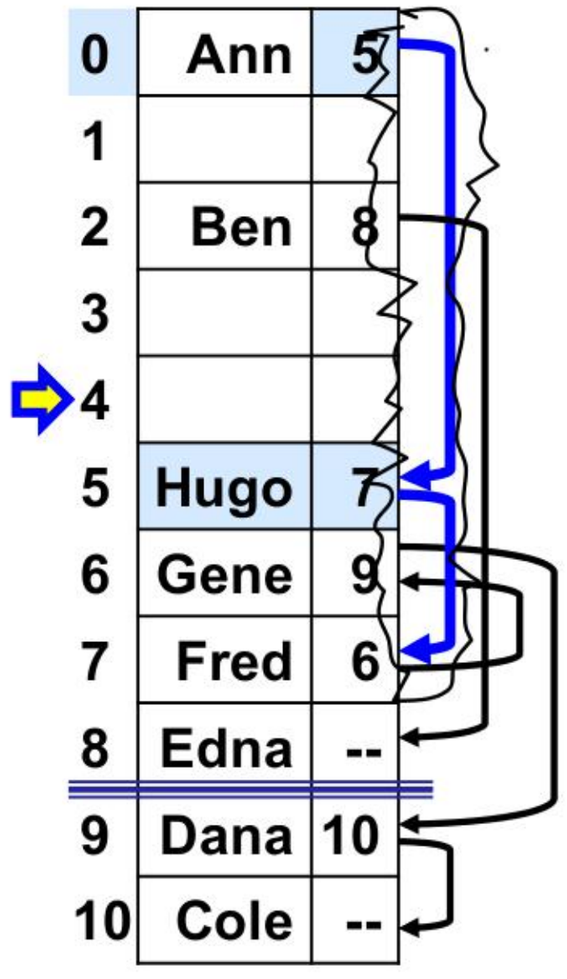
EICH (early insert coalesced hashing)

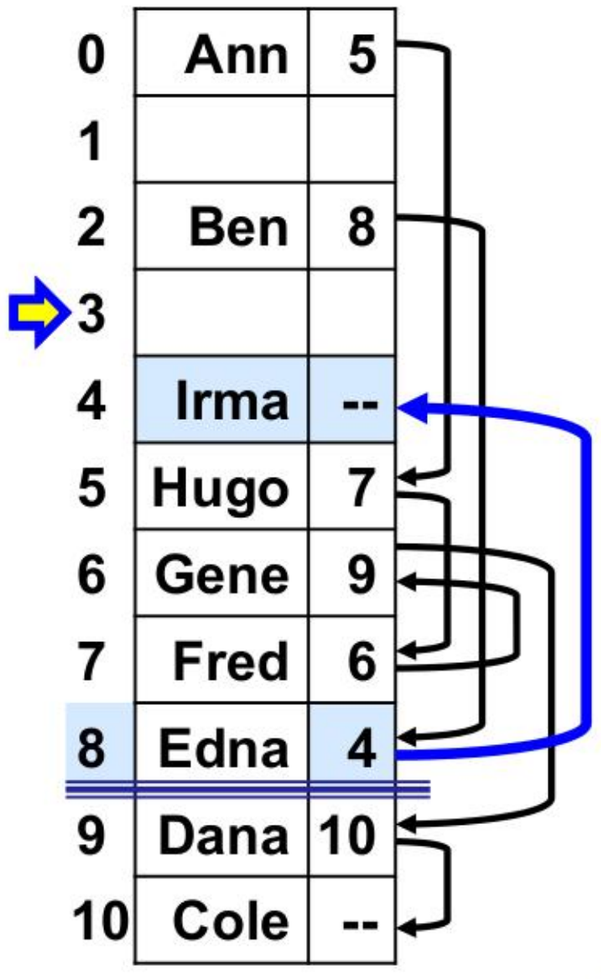


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

EICH (early insert coalesced hashing)

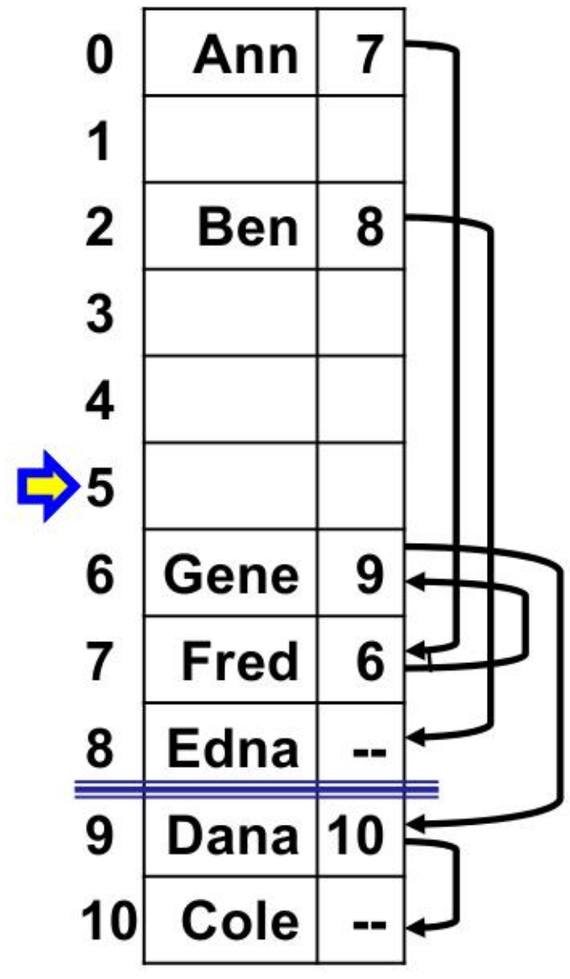


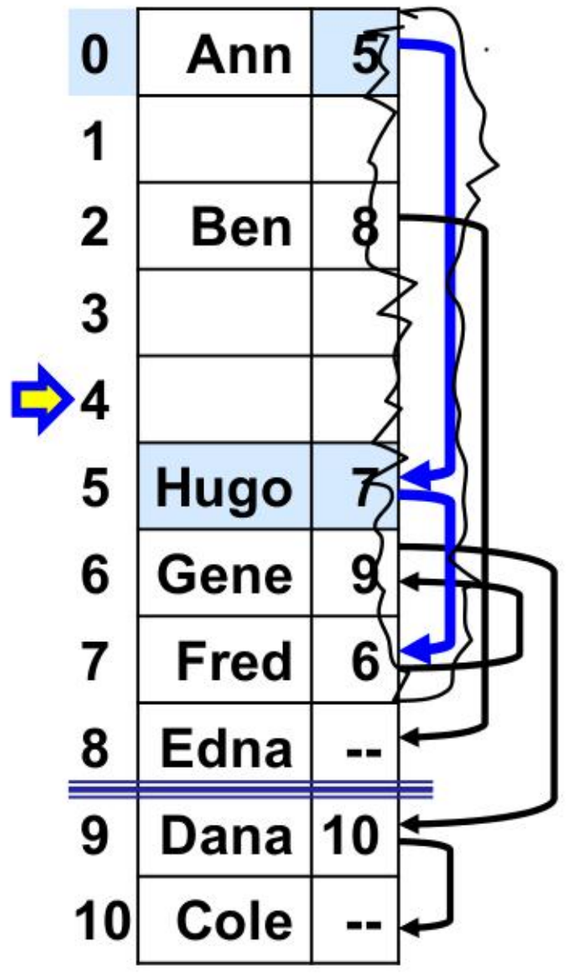


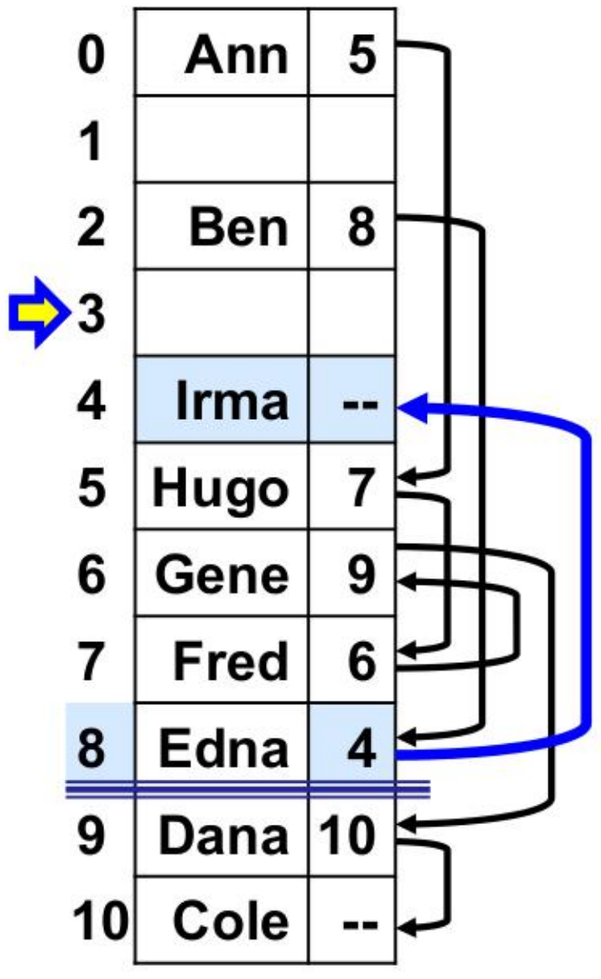


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

EICH (early insert coalesced hashing)







data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	7	0	8

VICH (variable insert coalesced hashing)

0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10		

0	Ann	10
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10	Cole	--

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	2	0	6

VICH (variable insert coalesced hashing)

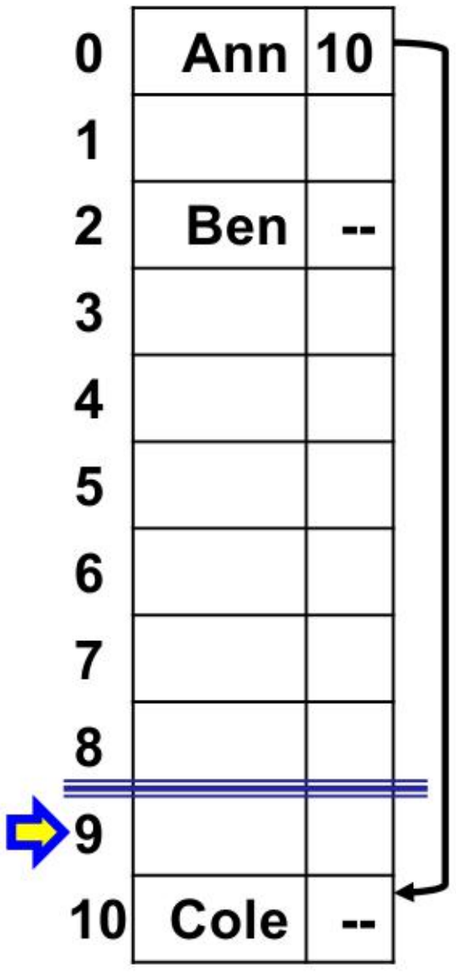
0	Ann	--
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		

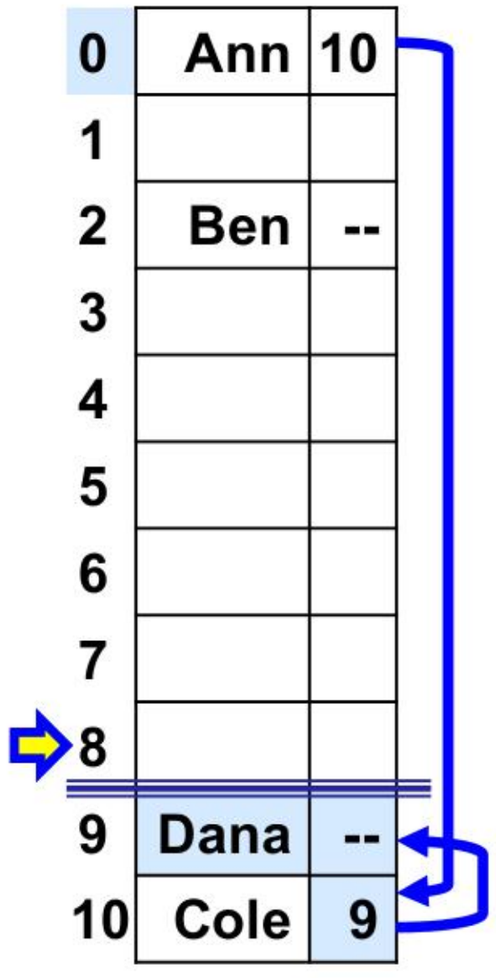
0	Ann	--
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10		

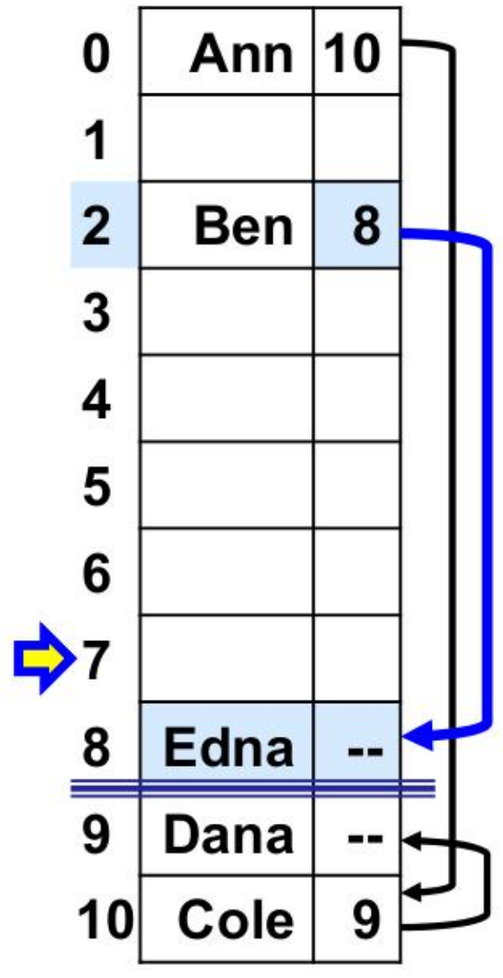
0	Ann	10
1		
2	Ben	--
3		
4		
5		
6		
7		
8		
9		
10	Cole	--

data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	2	0	6

VICH (variable insert coalesced hashing)

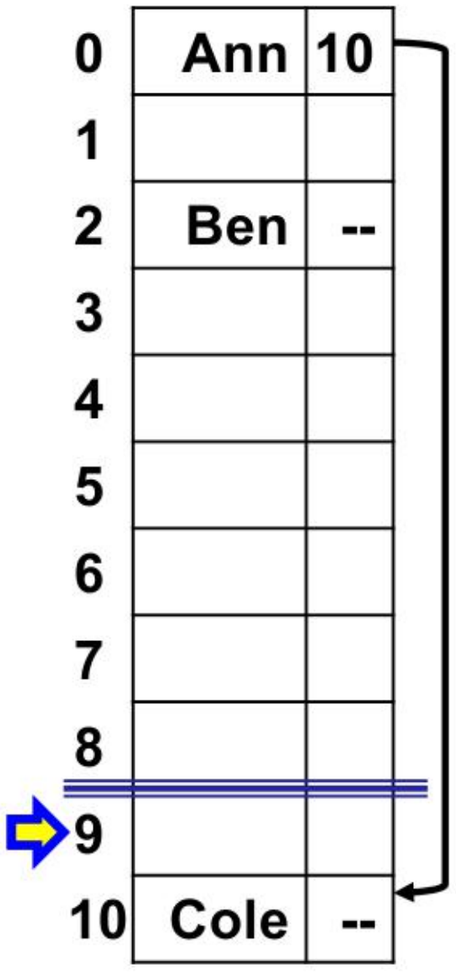


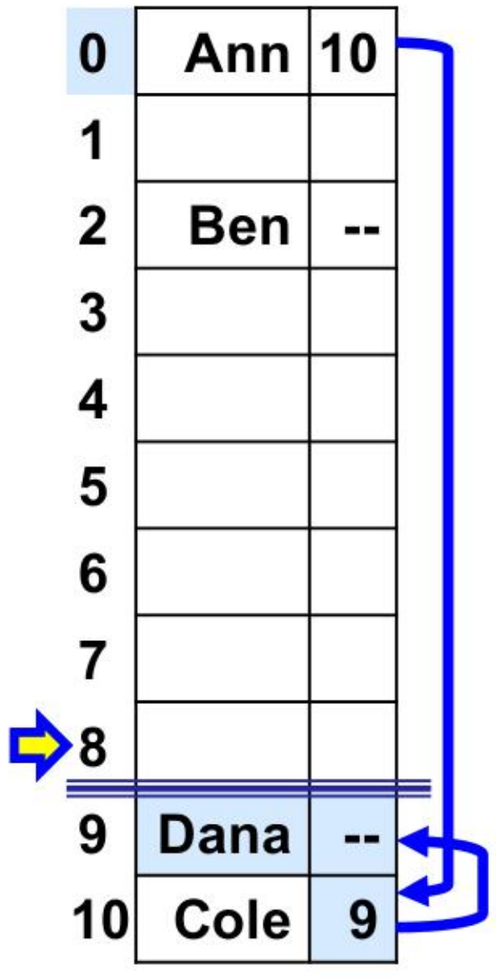


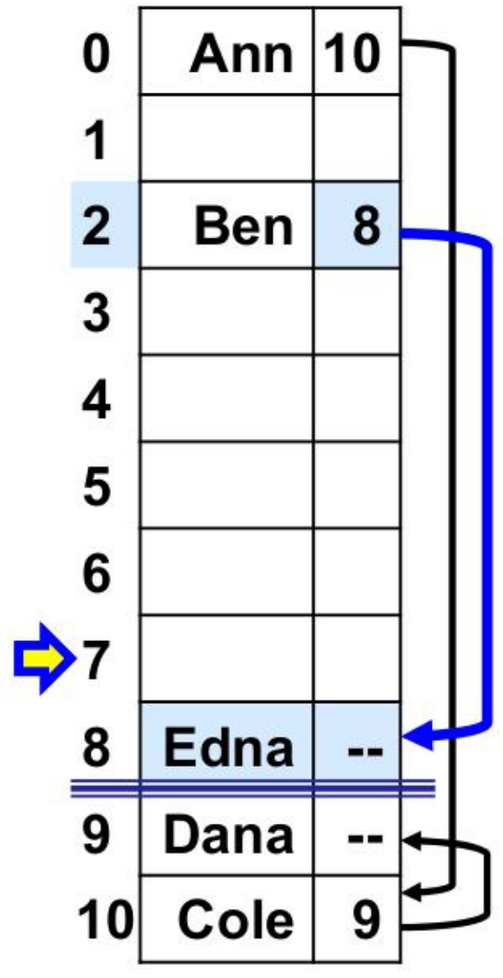


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	2	0	6

VICH (variable insert coalesced hashing)

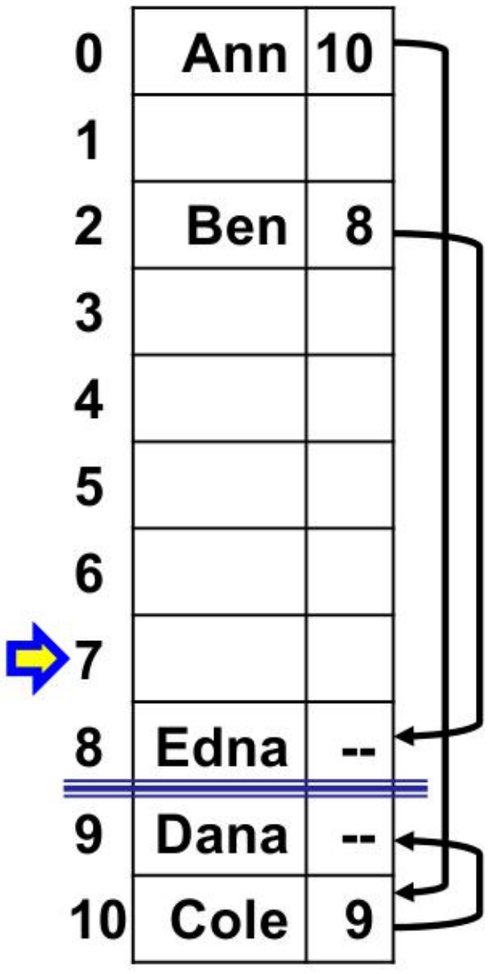


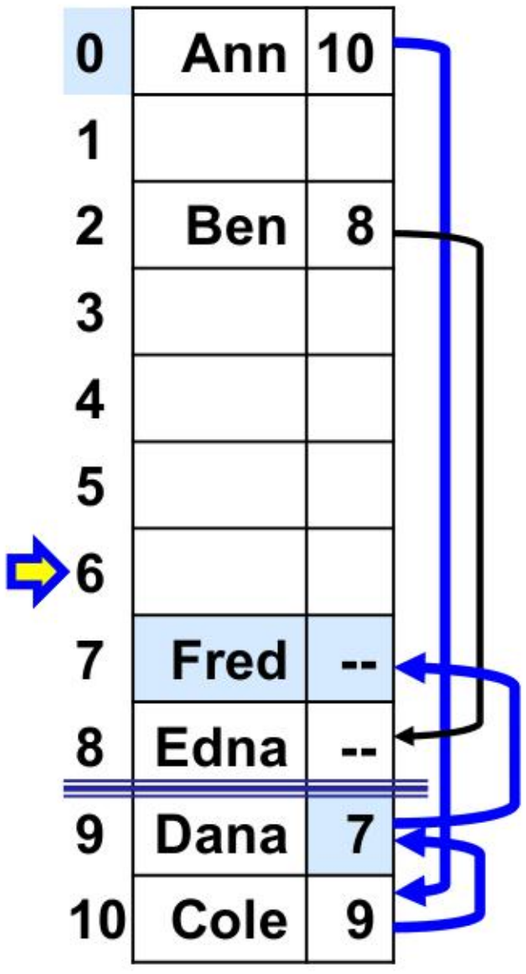


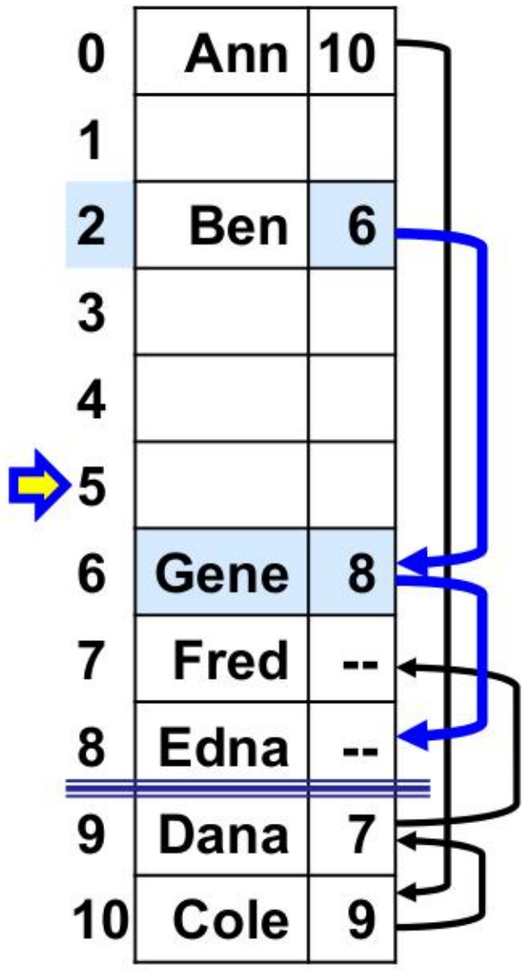


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	2	0	6

VICH (variable insert coalesced hashing)

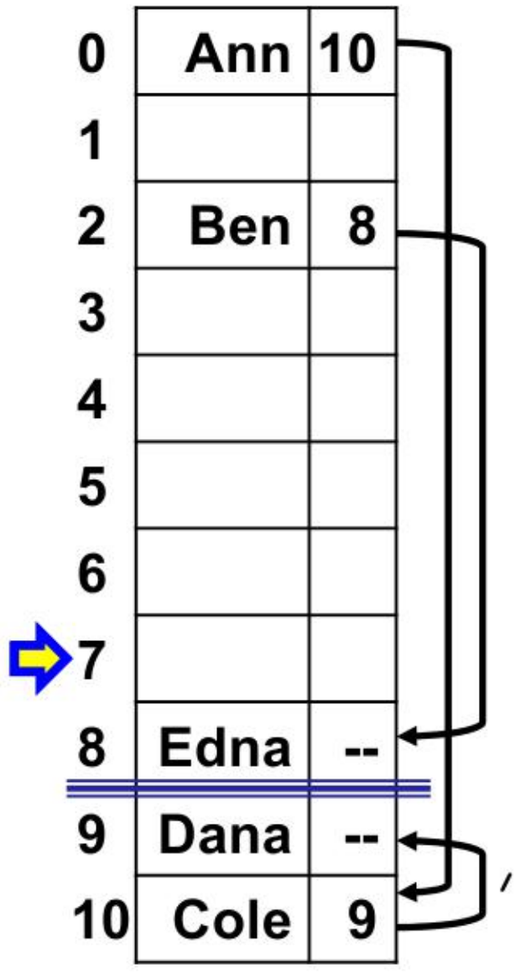


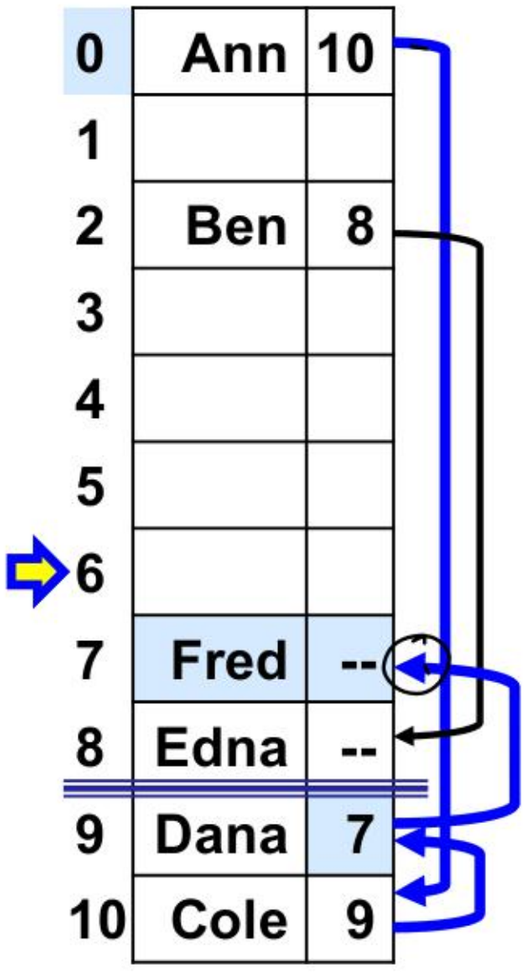


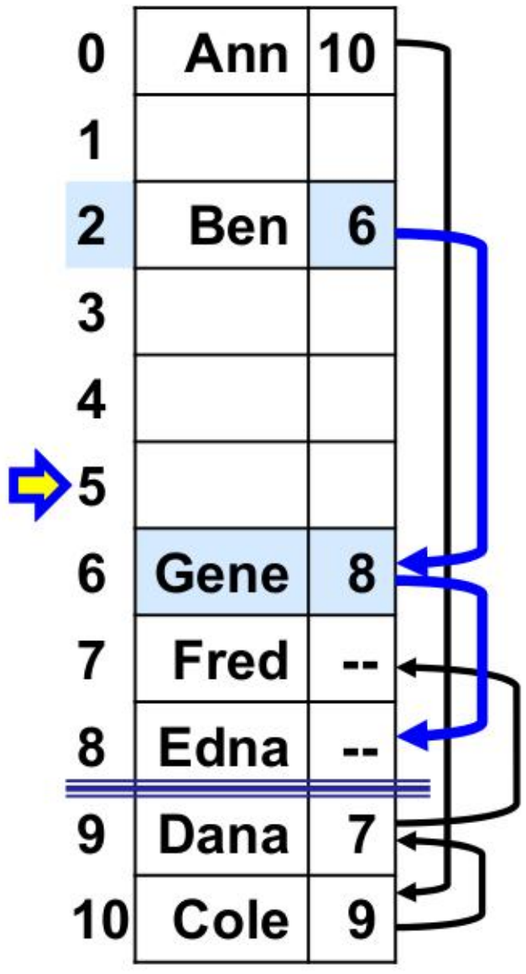


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	2	0	6

VICH (variable insert coalesced hashing)

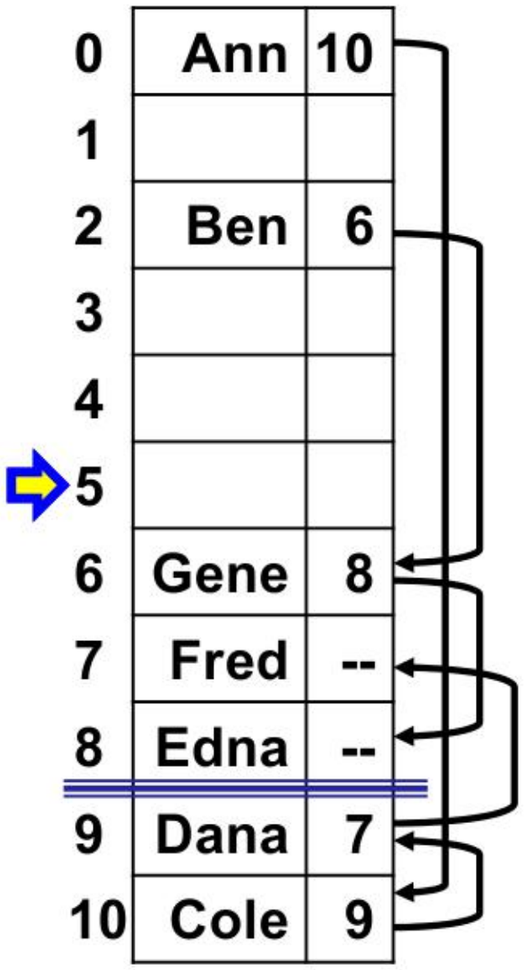


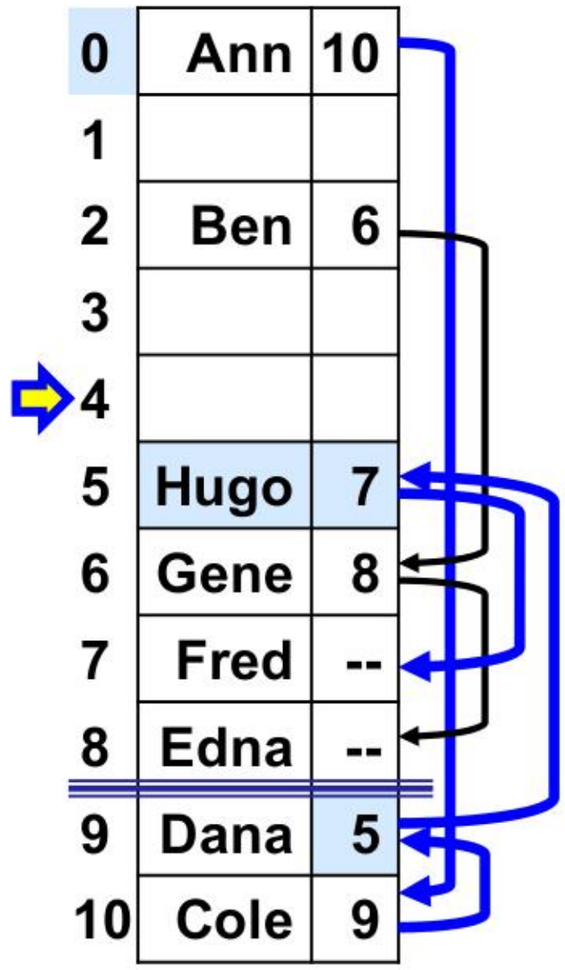


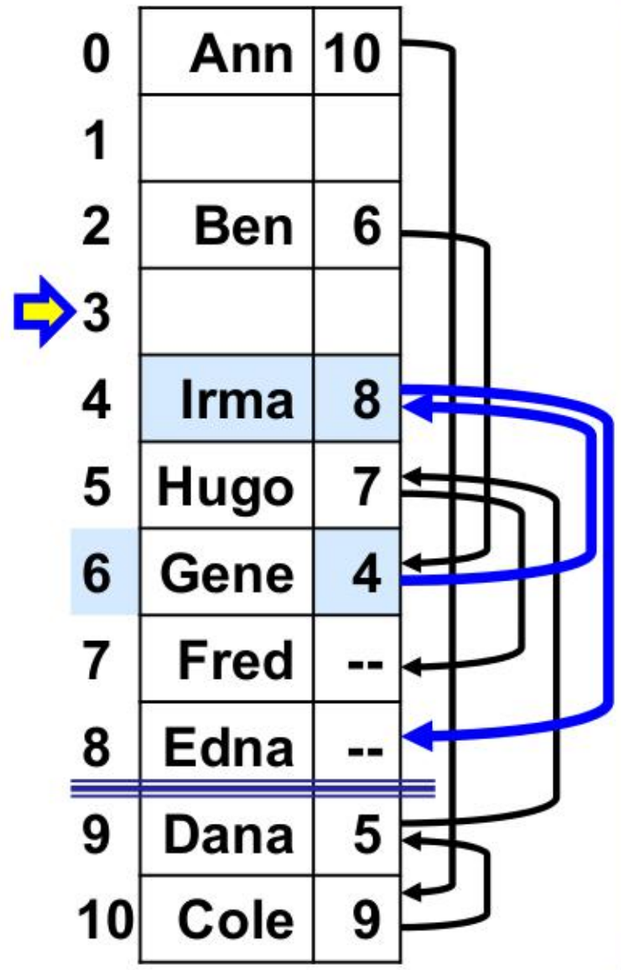


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	2	0	6

VICH (variable insert coalesced hashing)

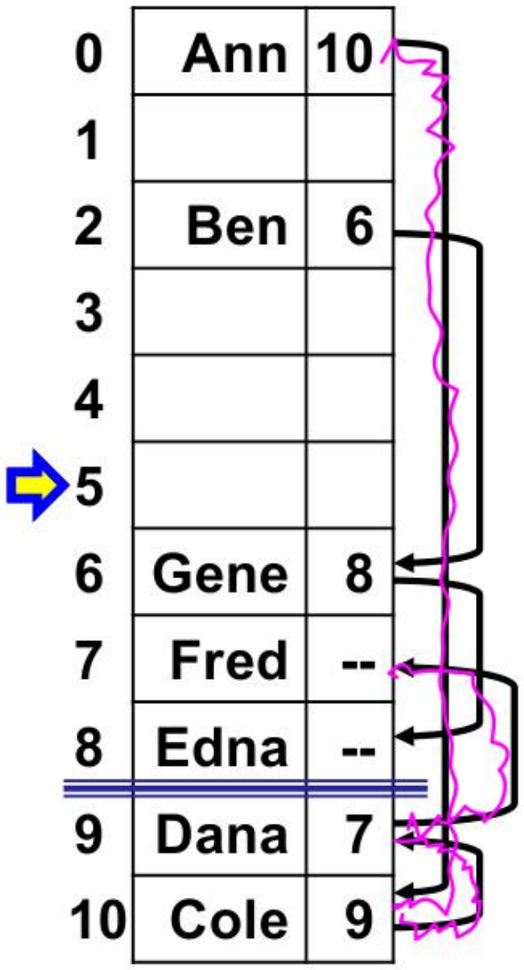


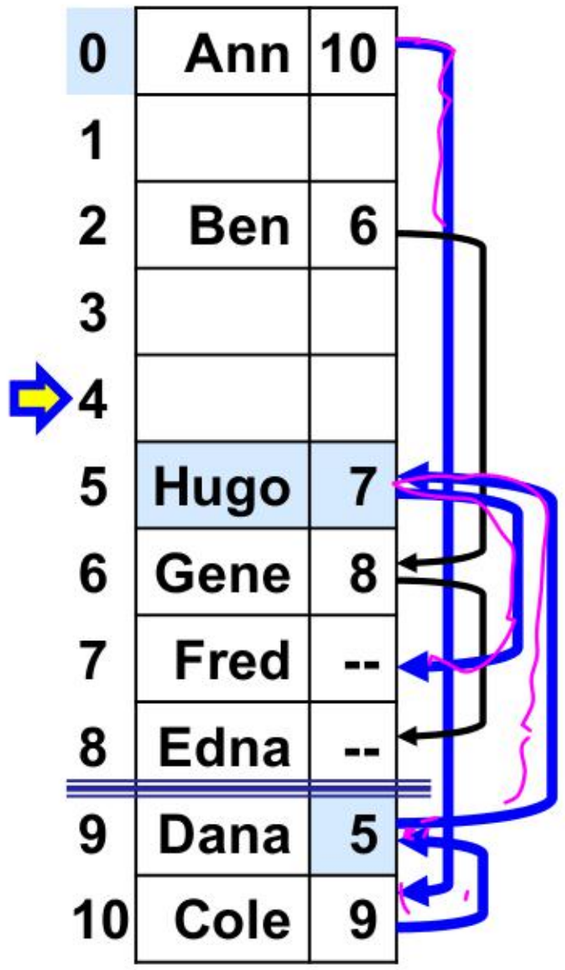


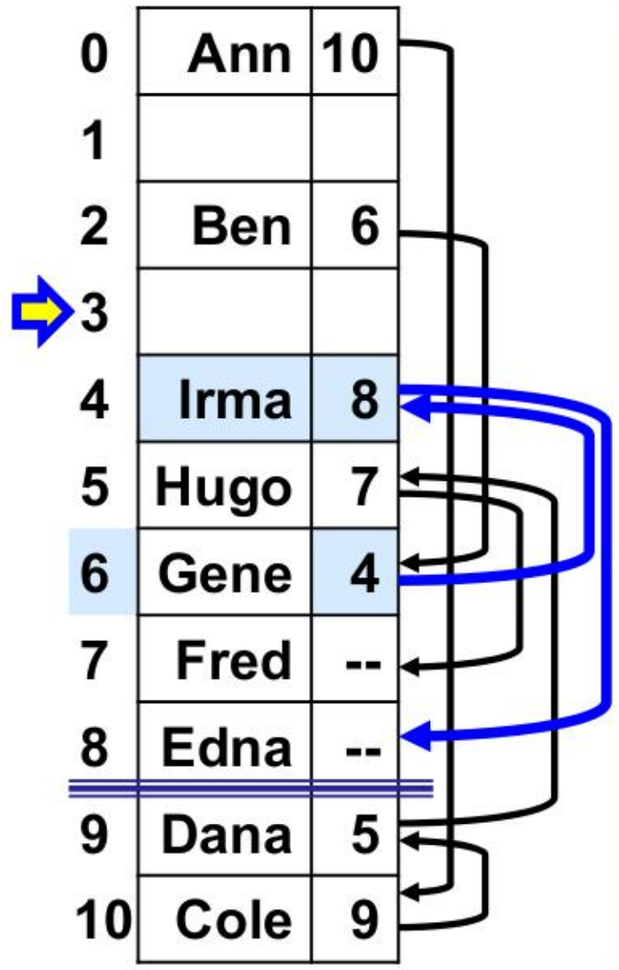


data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	2	0	6

VICH (variable insert coalesced hashing)







data	Ann	Ben	Cole	Dana	Edna	Fred	Gene	Hugo	Irma
h(data)	0	2	0	0	2	0	2	0	6

Algoritmizace

Hashing II

Jiří Vyskočil, Marko Genyg-Berezovskyj

2010

Srůstající hashování (coalesced hashing)

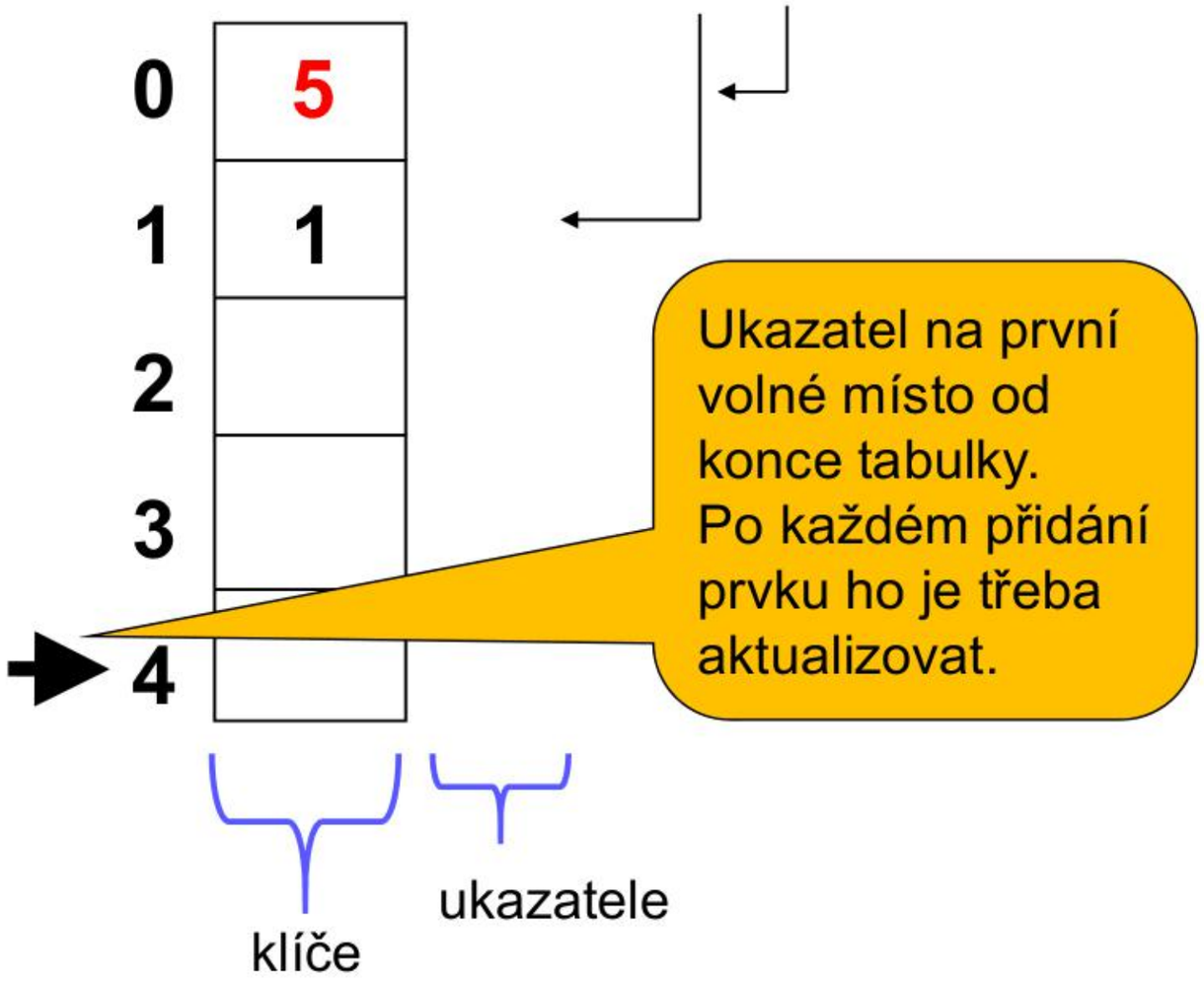
- Znám předem počet prvků (odhad)
- Z důvodů efektivity nechci ukazatele (mezi prvky). Na jednu pozici tabulky připadne právě jeden ukazatel.
- => kolizní prvky (synonyma) se musejí „nějak" ukládat přímo do hashovací tabulky. Ukazatele v tabulce umožňují procházet pouze prvky jedné skupiny. Nedochází zde k tak velkému propojování do clusterů jako v případě otevřeného hashování. Dochází zde pouze k tzv. srůstání.
  - standardní srůstající hashování
    - LISCH (late insert standard coalesced hashing)
    - EISCH (early insert standard coalesced hashing)
  - srůstající hashování s pomocnou pamětí
    - LICH (late insert coalesced hashing)
    - EICH (early insert coalesced hashing)
    - VICH (variable insert coalesced hashing)

Srůstající hashování (coalesced hashing)

- Znám předem počet prvků (odhad)
- Z důvodů efektivity nechci ukazatele (mezi prvky). Na jednu pozici tabulky připadne právě jeden ukazatel.
- => kolizní prvky (synonyma) se musejí „nějak" ukládat přímo do hashovací tabulky. Ukazatele v tabulce umožňují procházet pouze prvky jedné skupiny. Nedochází zde k tak velkému propojování do clusterů jako v případě otevřeného hashování. Dochází zde pouze k tzv. srůstání.
  - standardní srůstající hashování
    - LISCH (late insert standard coalesced hashing)
    - EISCH (early insert standard coalesced hashing)
  - srůstající hashování s pomocnou pamětí
    - LICH (late insert coalesced hashing)
    - EICH (early insert coalesced hashing)
    - VICH (variable insert coalesced hashing)

Standardní srůstající hashování - LISCH

- $h(k) = k \bmod 5$
- posloupnost: 1, 5, 21, 10, 15

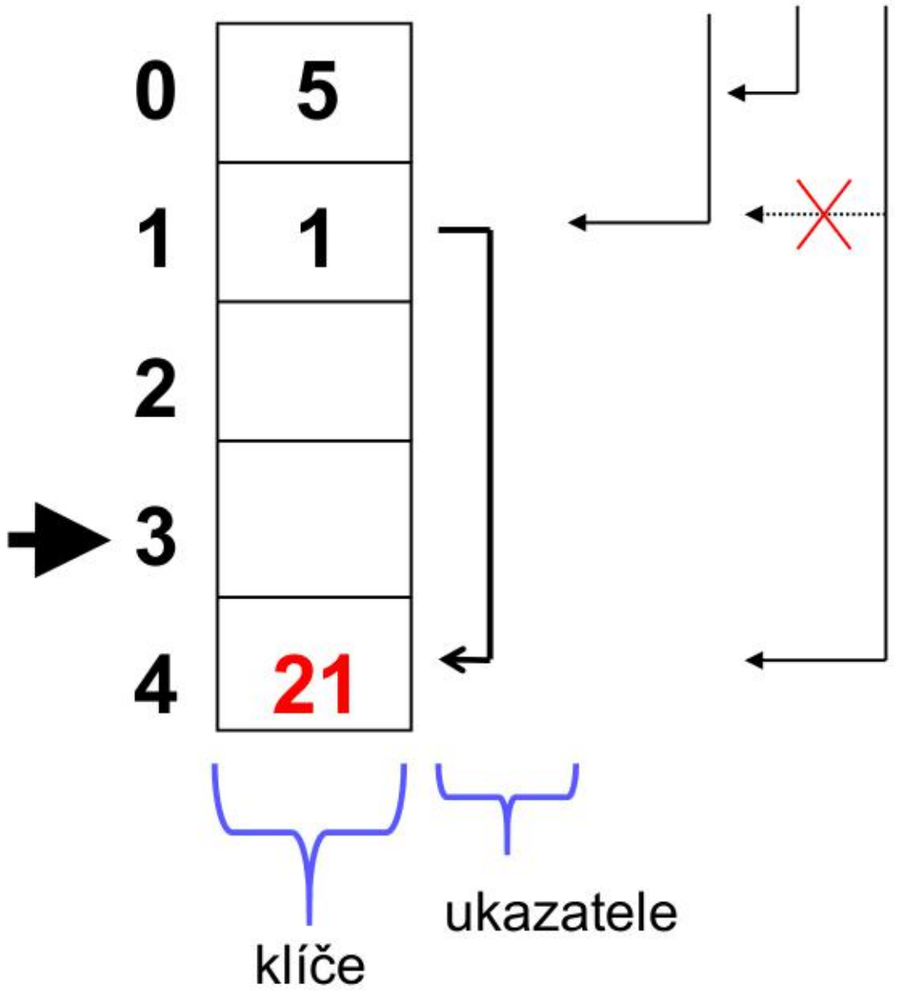


Postup:

1. $i = h(k)$;
2. Prohledej řetězec začínající na místě i a pokud nenajdeš k, přidej ho do tabulky na první volné místo od konce tabulky a připoj ho do řetězce na poslední místo.

Standardní srůstající hashování - LISCH

- $h(k) = k \bmod 5$
- posloupnost: 1, 5, 21, 10, 15

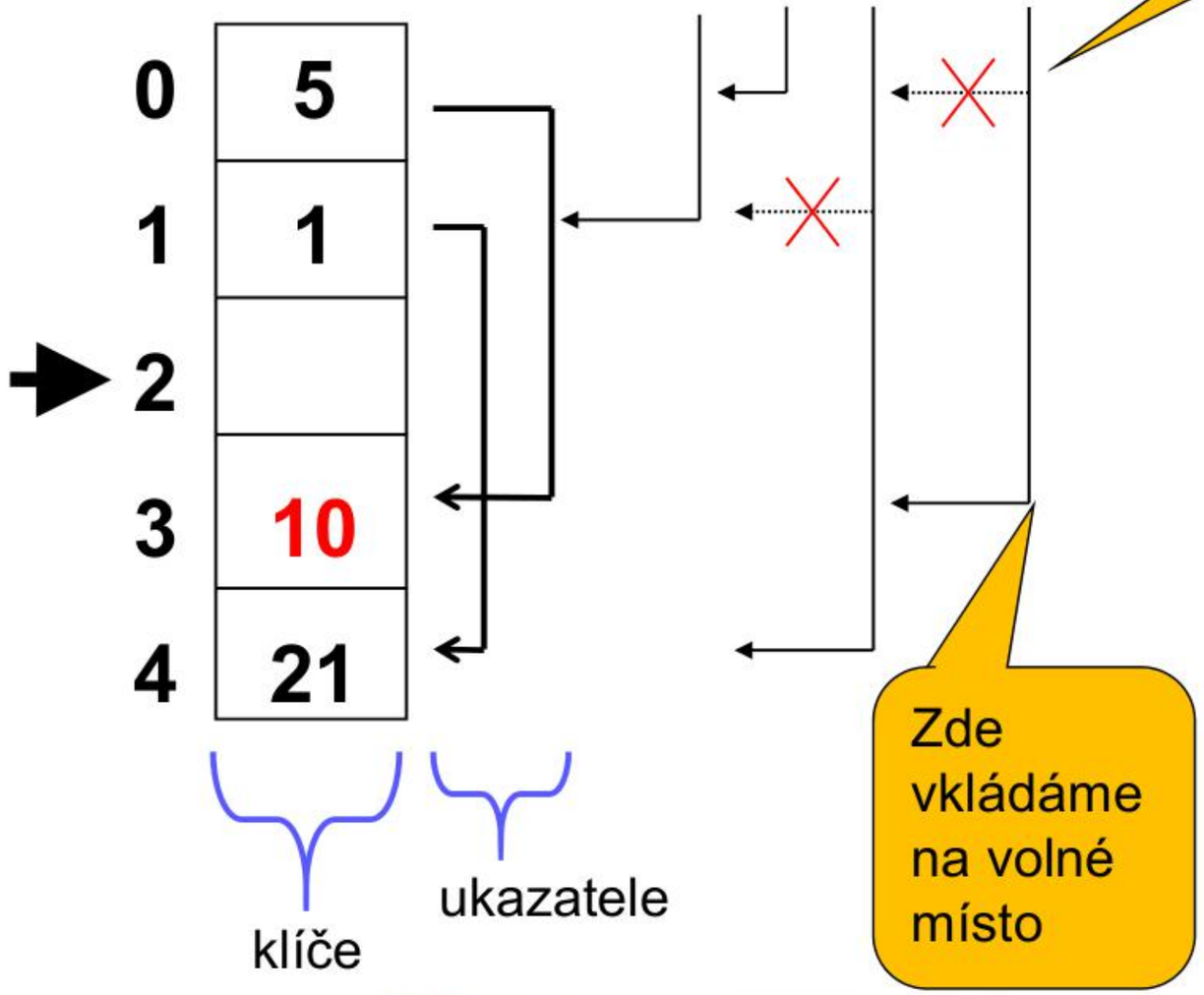


Postup:

1. $i = h(k)$;
2. Prohledej řetězec začínající na místě i a pokud nenajdeš k, přidej ho do tabulky na první volné místo od konce tabulky a připoj ho do řetězce na poslední místo.

Standardní srůstající hashování - LISCH

- $h(k)$ = k mod 5
- posloupnost: 1, 5, 21, 10, 15



Zde procházíme řetězec prvků.

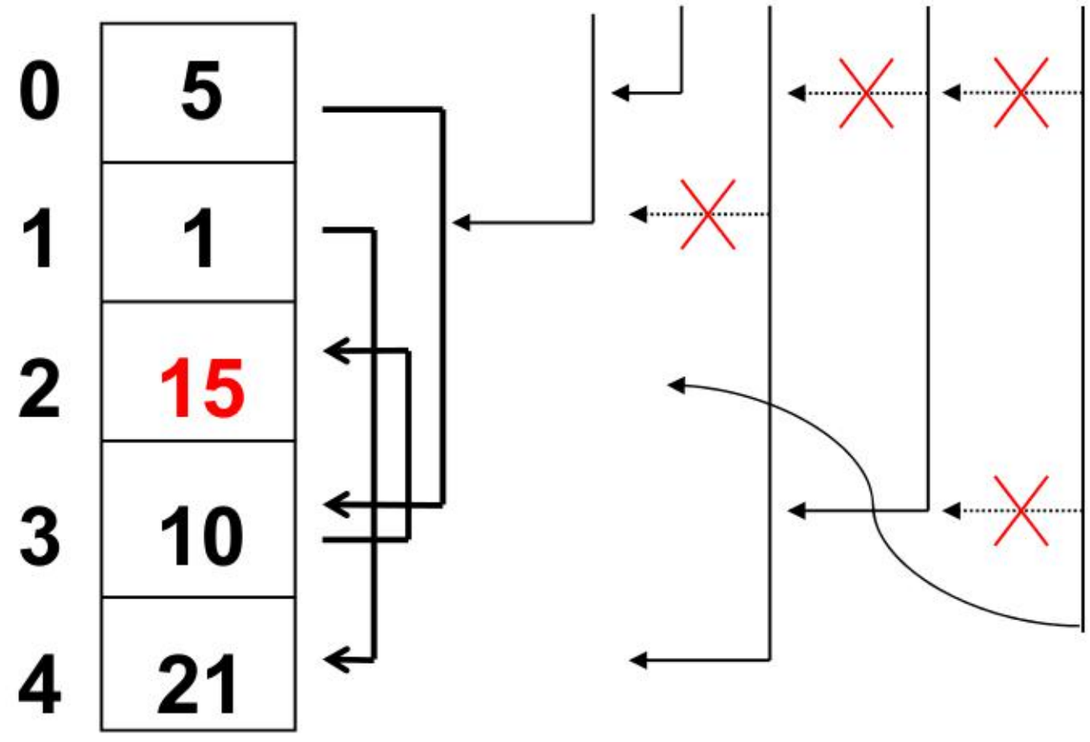
Zde vkládáme na volné místo

Postup:

1. $i = h(k)$;
2. Prohledej řetězec začínající na místě i a pokud nenajdeš k, přidej ho do tabulky na první volné místo od konce tabulky a připoj ho do řetězce na poslední místo.

Standardní srůstající hashování - LISCH

- $h(k) = k \bmod 5$
- posloupnost: 1, 5, 21, 10, 15



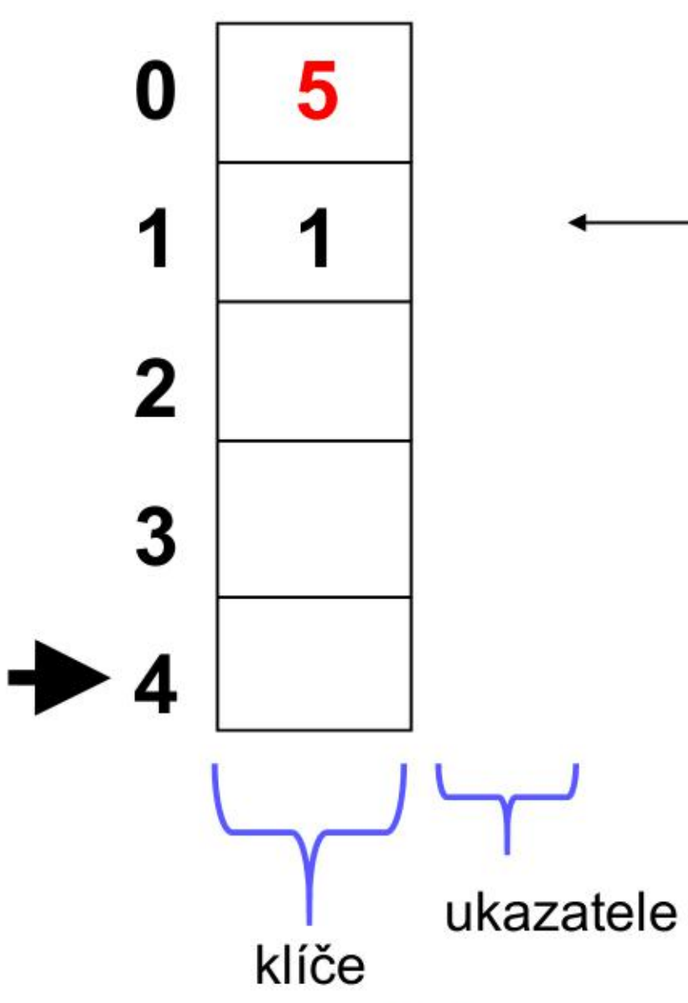
Postup:

1. $i = h(k)$;
2. Prohledej řetězec začínající na místě i a pokud nenajdeš k, přidej ho do tabulky na první volné místo od konce tabulky a připoj ho do řetězce na poslední místo.

➡ Tabulka je zaplněna.

Standardní srůstající hashování - EISCH

- $h(k) = k \bmod 5$
- posloupnost: 1, 5, 21, 10, 15

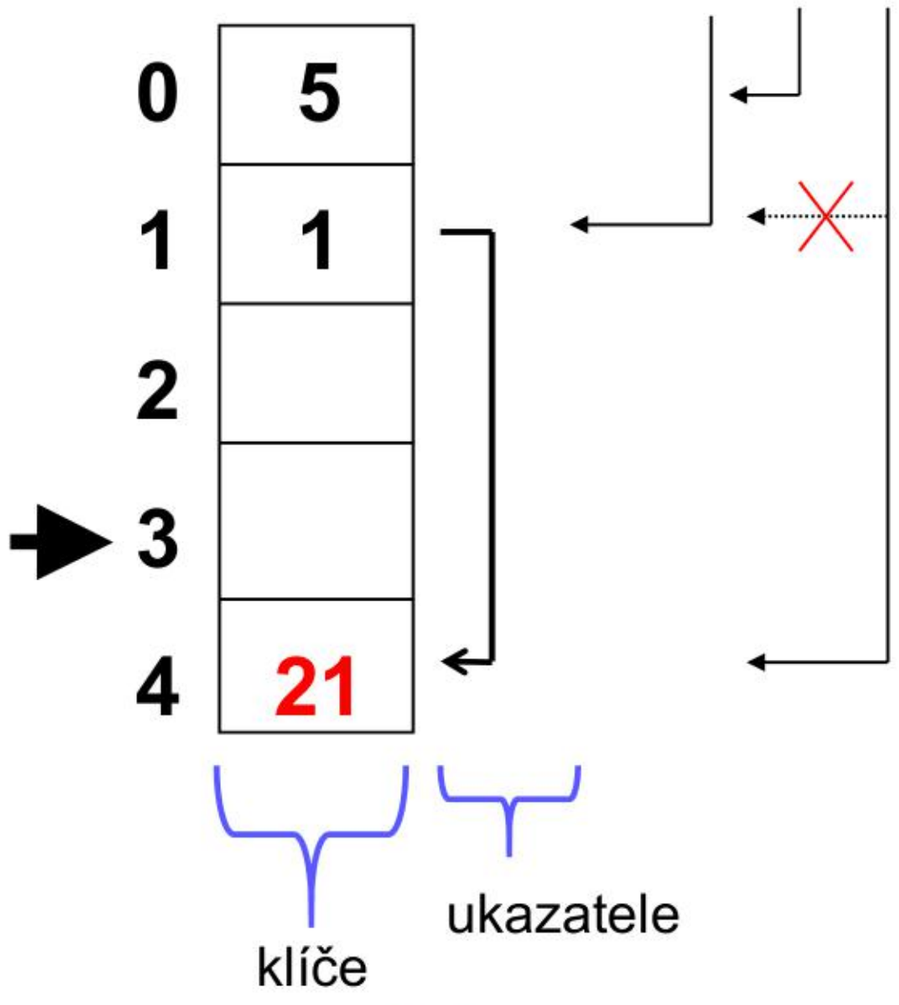


Postup:

1. $i = h(k)$;
2. Prohledej řetězec začínající na místě i a pokud nenajdeš k, přidej ho do tabulky na první volné místo od konce tabulky a připoj ho do řetězce za 1. místo.

Standardní srůstající hashování - EISCH

- $h(k) = k \bmod 5$
- posloupnost: 1, 5, 21, 10, 15

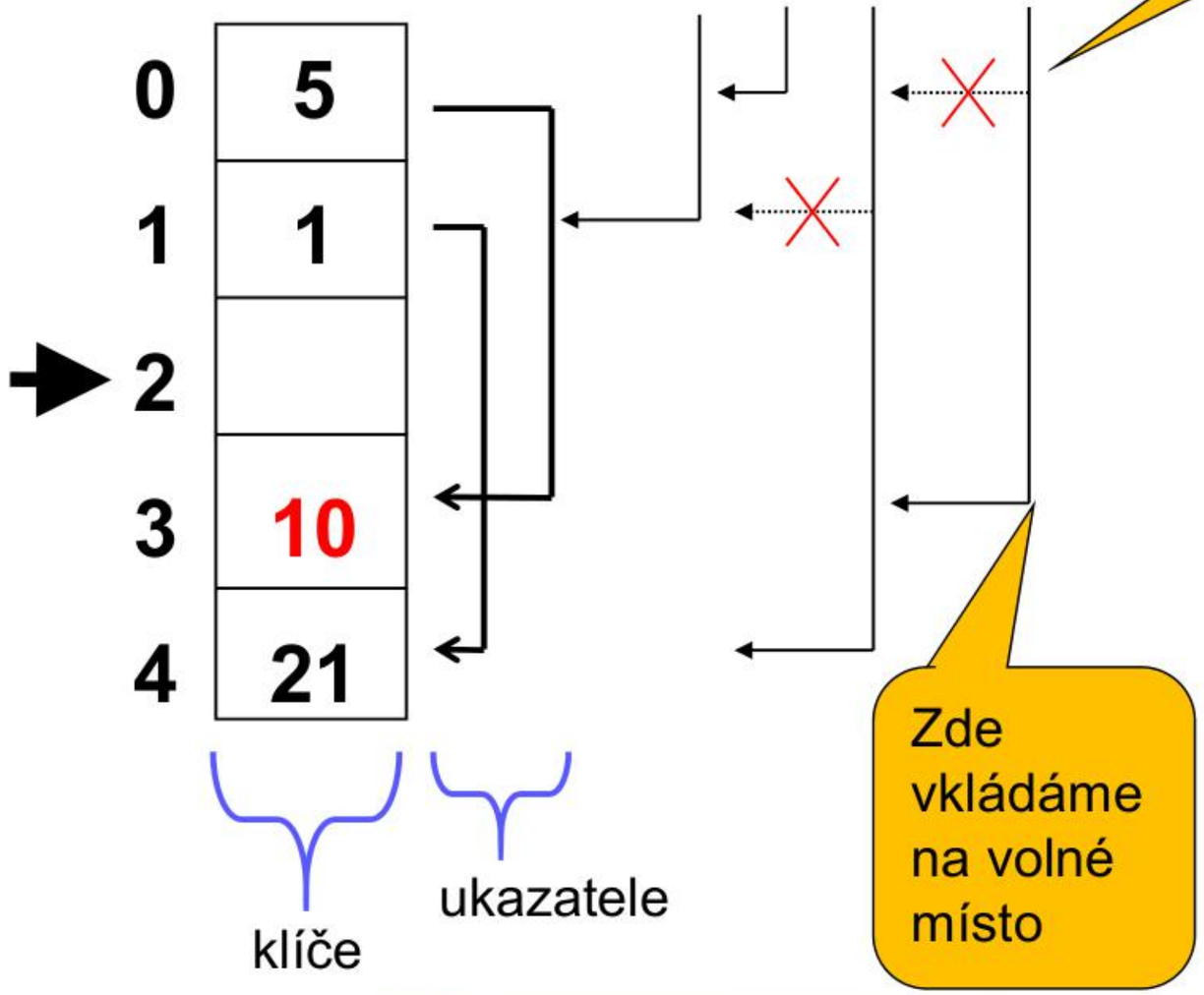


Postup:

1. $i = h(k)$;
2. Prohledej řetězec začínající na místě i a pokud nenajdeš k, přidej ho do tabulky na první volné místo od konce tabulky a připoj ho do řetězce za 1. místo.

Standardní srůstající hashování - EISCH

- $h(k) = k \bmod 5$
- posloupnost: 1, 5, 21, 10, 15



Zde procházíme řetězec prvků.

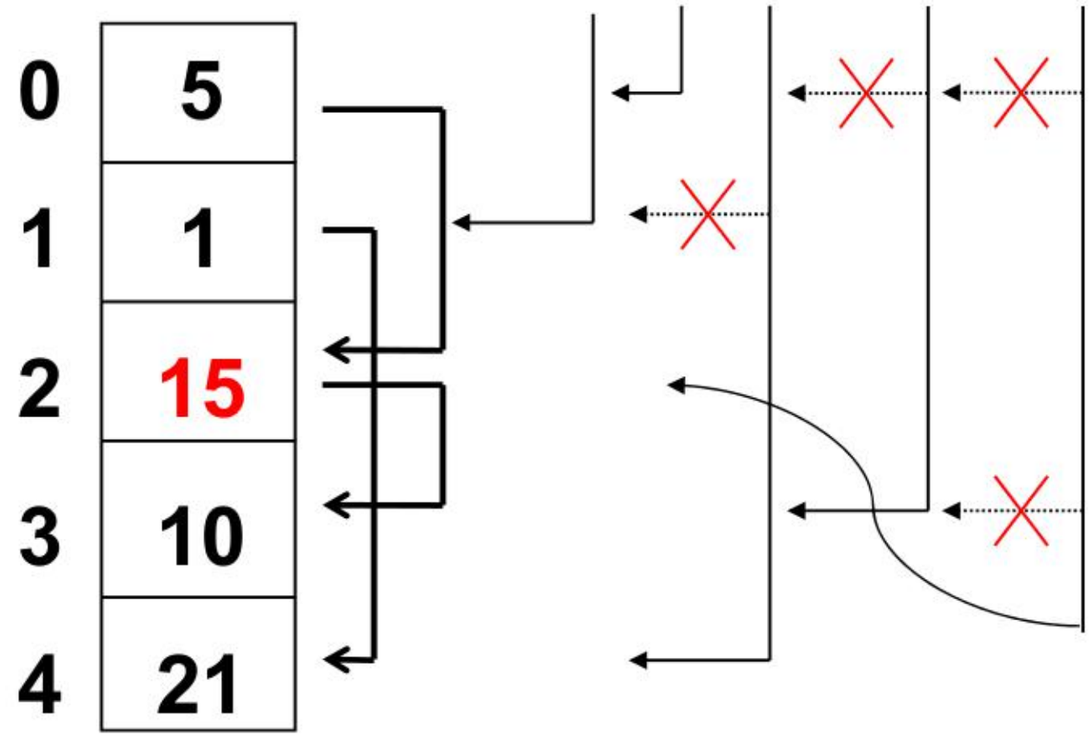
Zde vkládáme na volné místo

Postup:

1. $i = h(k)$;
2. Prohledej řetězec začínající na místě i a pokud nenajdeš k, přidej ho do tabulky na první volné místo od konce tabulky a připoj ho do řetězce za 1. místo.

Standardní srůstající hashování - EISCH

- $h(k) = k \bmod 5$
- posloupnost: 1, 5, 21, 10, 15

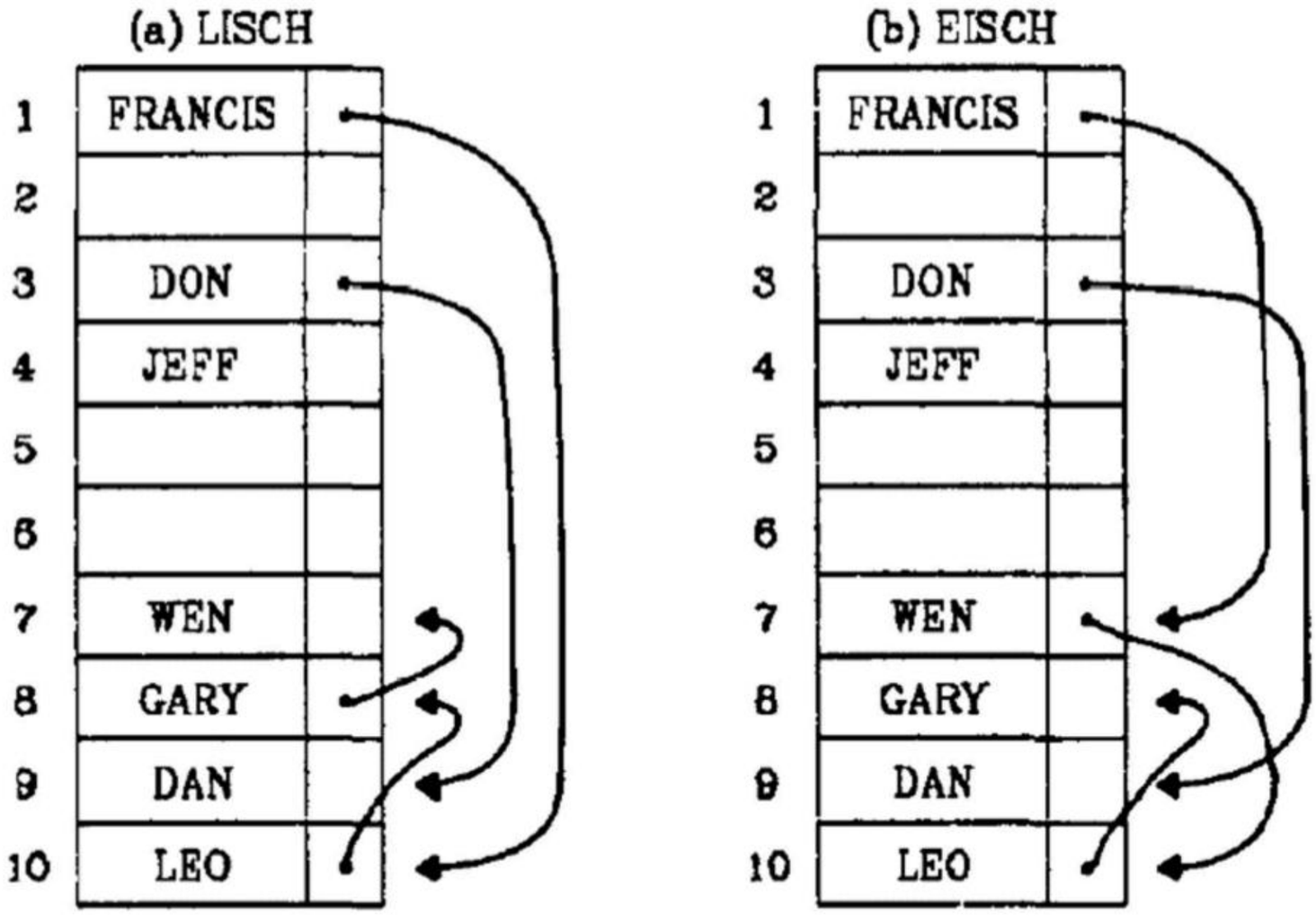


Postup:

1. $i = h(k)$;
2. Prohledej řetězec začínající na místě i a pokud nenajdeš k, přidej ho do tabulky na první volné místo od konce tabulky a připoj ho do řetězce za 1. místo.

➡ Tabulka je zaplněna.

Standardní srůstající hashování – LISCH, EISCH

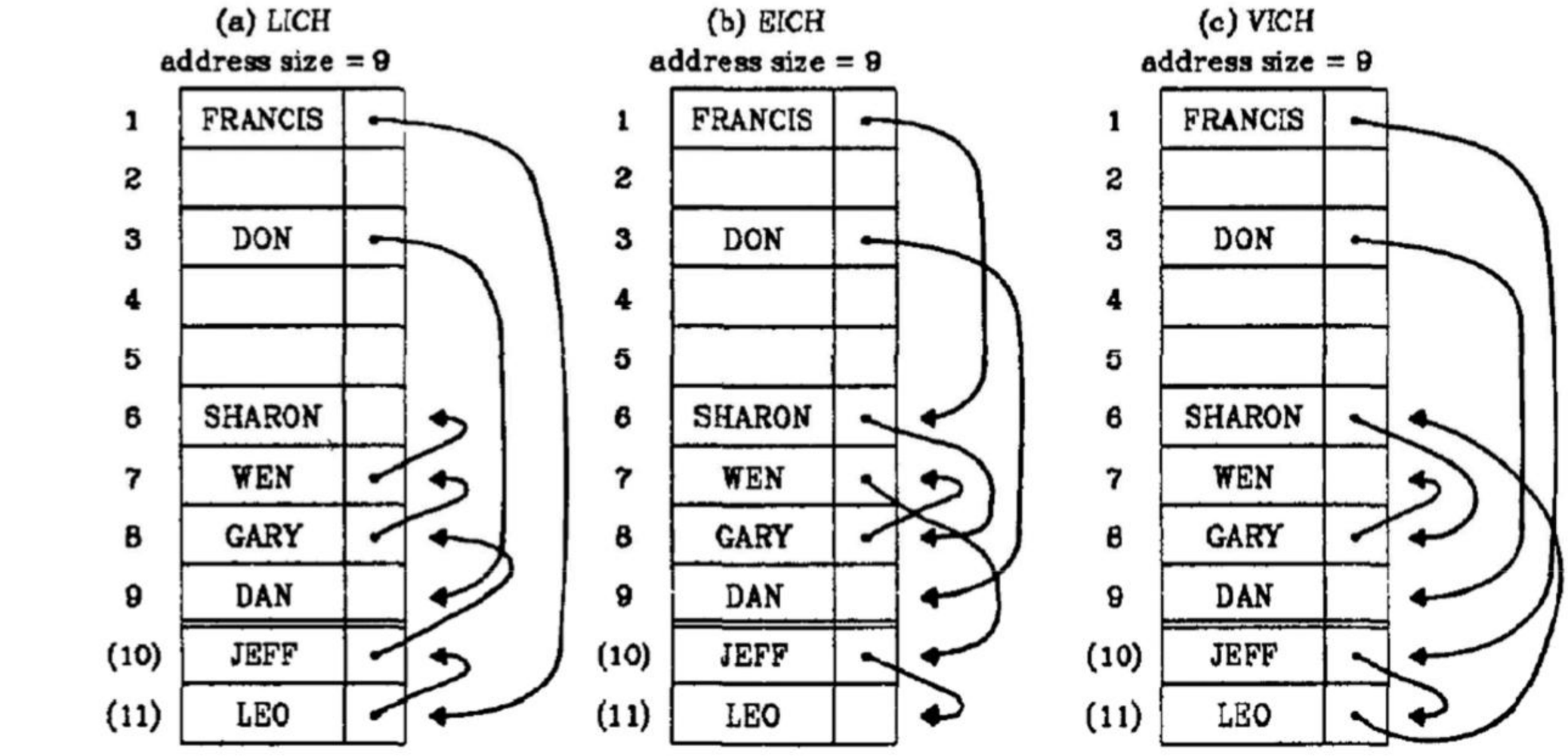


Keys:		FRANCIS	DON	LEO	JEFF	DAN	GARY	WEN
Hash Addresses:	(a)(b)	1	3	1	4	3	10	1

Srůstající hashování s pomocnou pamětí

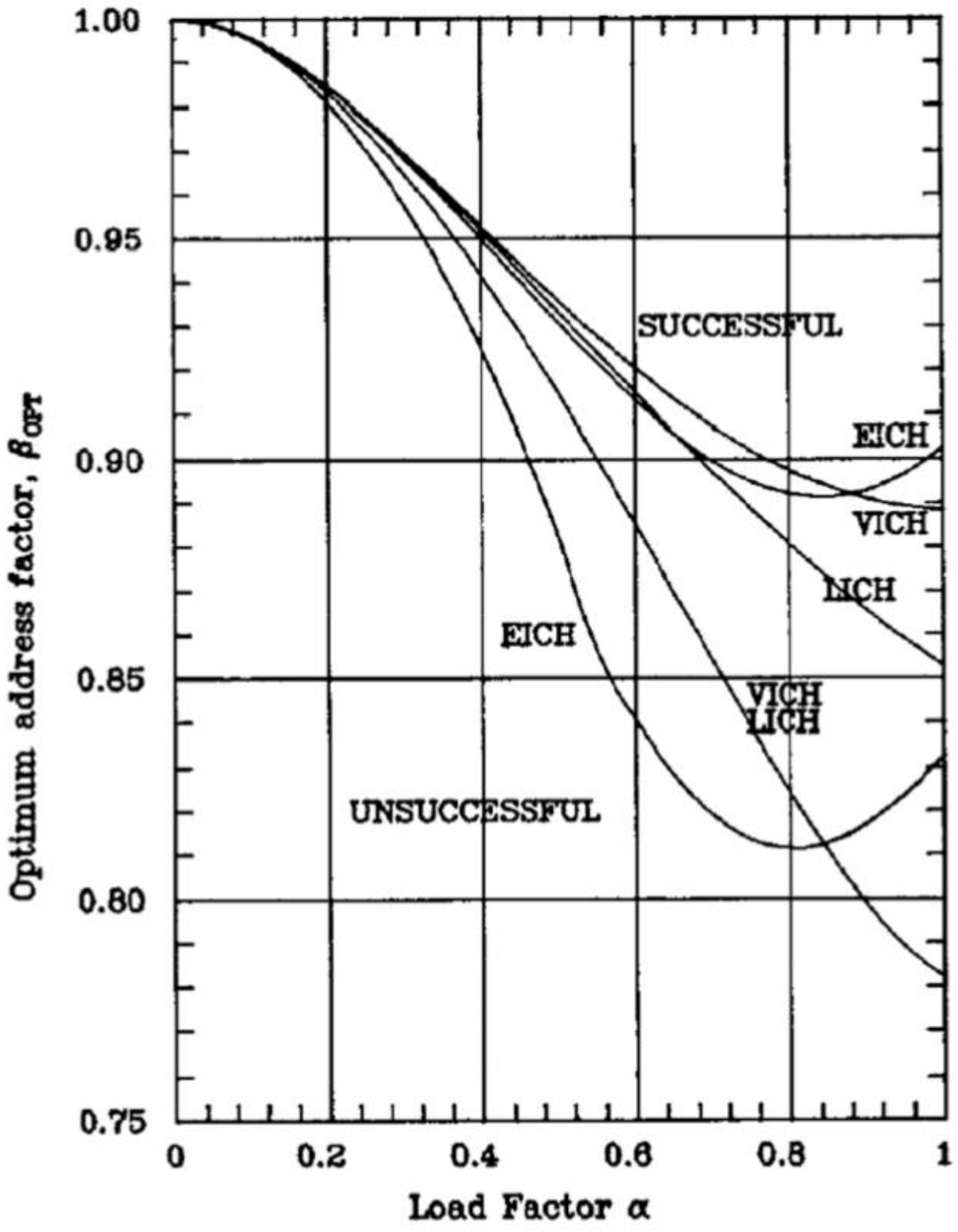
- Pro snížení srůstání a tedy zvýšení efektivity hashování se tabulka rozšiřuje o pomocnou paměť - tzv. sklep (cellar).
- Sklep je místo na konci tabulky, které není adresovatelné hashovací funkcí (má ale stejnou strukturu jako celá hashovací tabulka).
- Algoritmy LICH a EICH jsou analogické varianty algoritmů LISCH a EISCH s přidáním sklepa.
- Algoritmus VICH (variable insert coalesced hashing) připojuje prvek na konec řetězce, pokud řetězec končí ve sklepě, jinak na místo, kde řetězec opustil sklep.

Srůstající hashování s pomocnou pamětí



Keys:	FRANCIS	DON	LEO	JEFF	DAN	GARY	WEN	SHARON
Hash Addresses:	1	3	1	1	3	1	8	1

Srůstající hashování s pomocnou pamětí



α – faktor naplnění (load factor)
$\alpha = N/M'$

β – faktor adresování (address factor)
$\beta = M/M'$

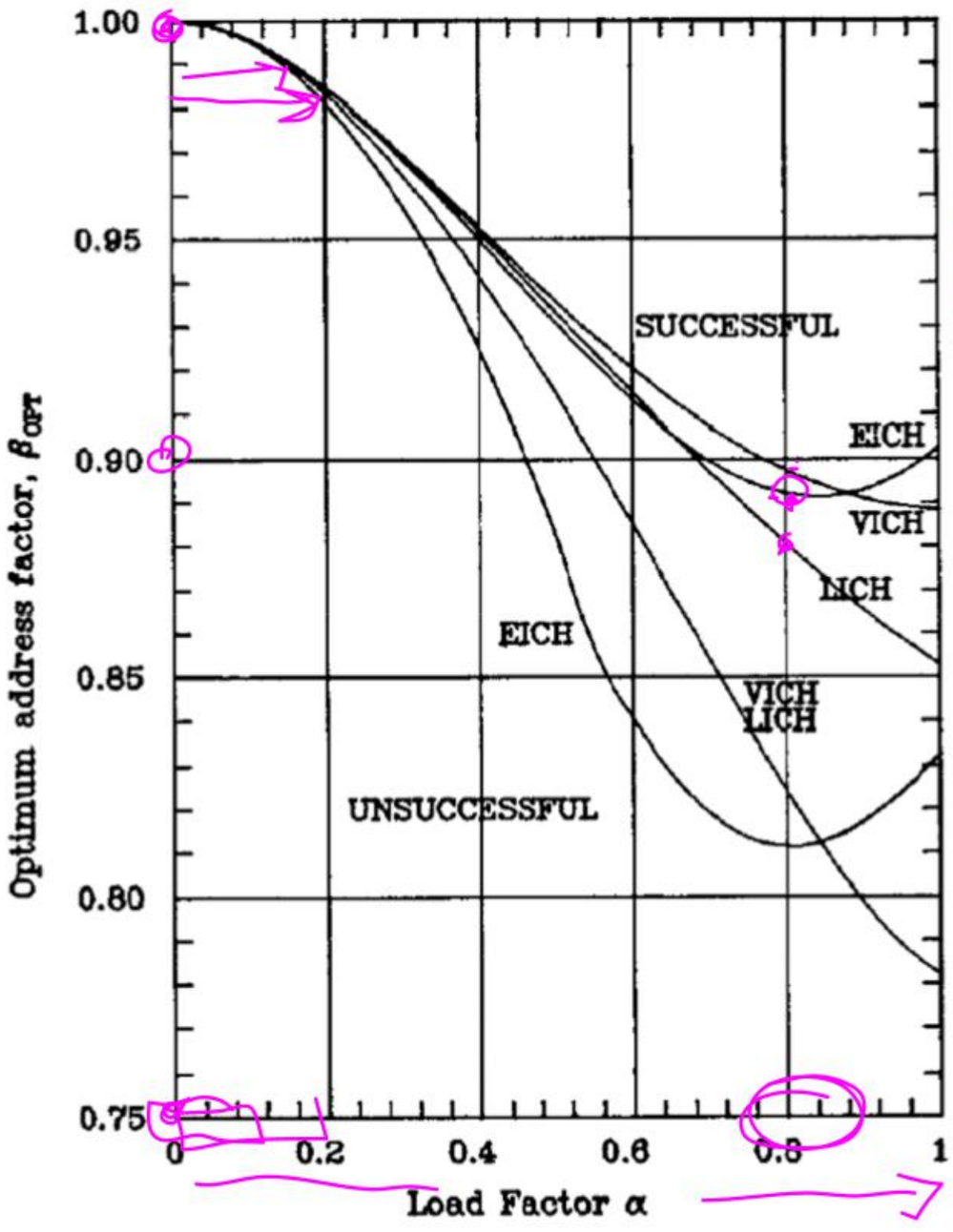
$K = M' - M$ = velikost sklepa

N – počet vložených prvků
M' – počet míst v hashovací tabulce
M – počet míst adresovatelných hashovací funkcí

Křivky představují optimální volbu β
v závislosti na daném α při operaci FIND.

Případy:
SUCCESSFUL - klíč je v tabulce
UNSUCCESSFUL - klíč není v tabulce

Srůstající hashování s pomocnou pamětí



α – faktor naplnění (load factor)
$\alpha = N/M'$

β – faktor adresování (address factor)
$\beta = M/M'$

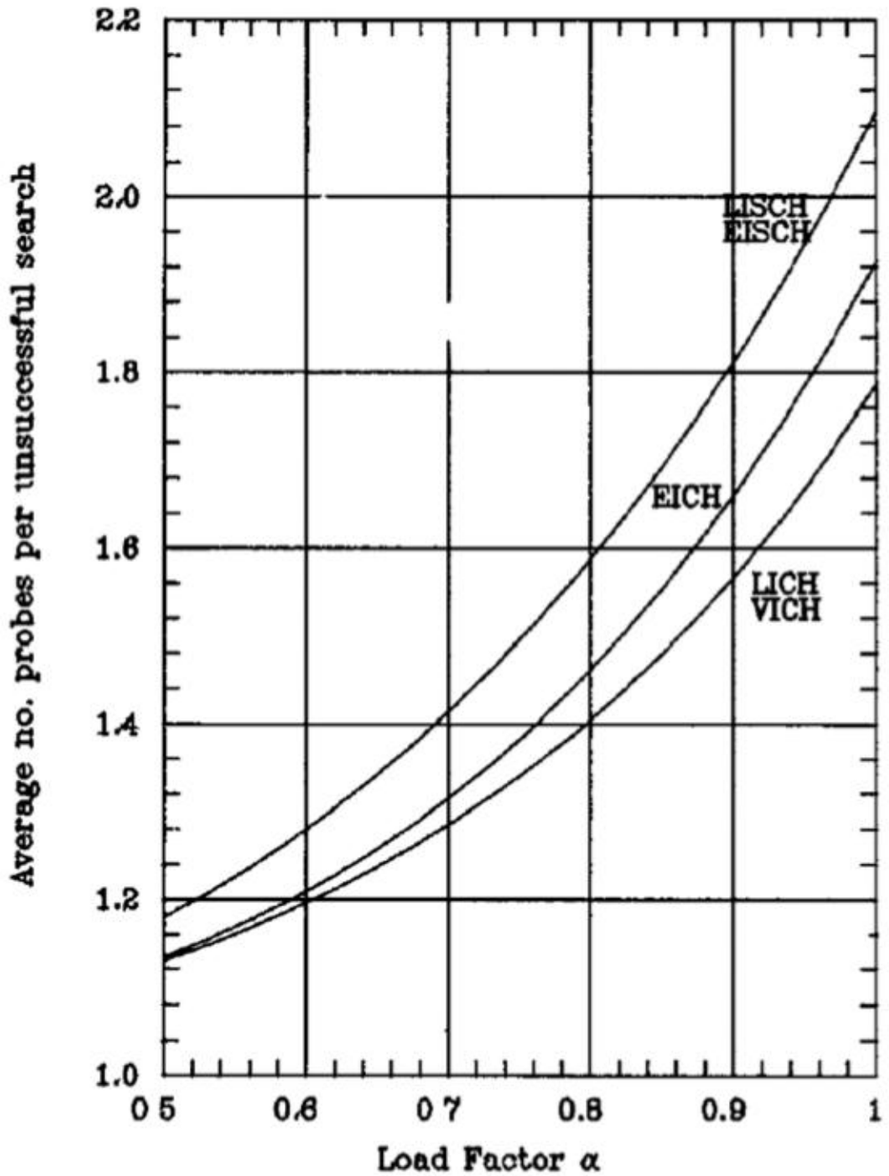
$K = M' - M$ = velikost sklepa

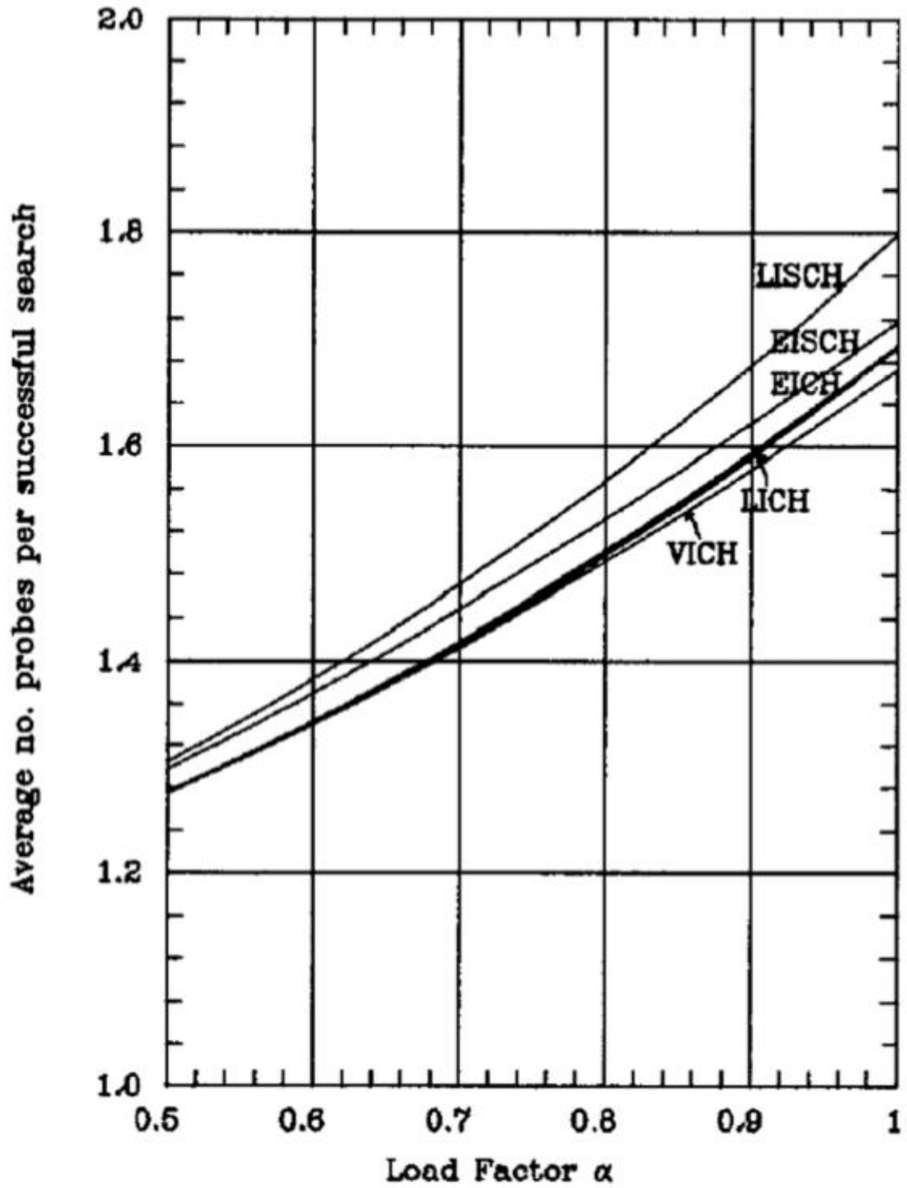
N – počet vložených prvků
M' – počet míst v hashovací tabulce
M – počet míst adresovatelných hashovací funkcí

Křivky představují optimální volbu β v závislosti na daném α při operaci FIND.

Případy:
SUCCESSFUL - klíč je v tabulce
UNSUCCESSFUL - klíč není v tabulce

Celkové srovnání srůstajícího hashování





- S použitím sklepa vychází nejlépe VICH. Doporučená velikost β je 0,86.
- Bez sklepa vychází nejlépe EISCH.

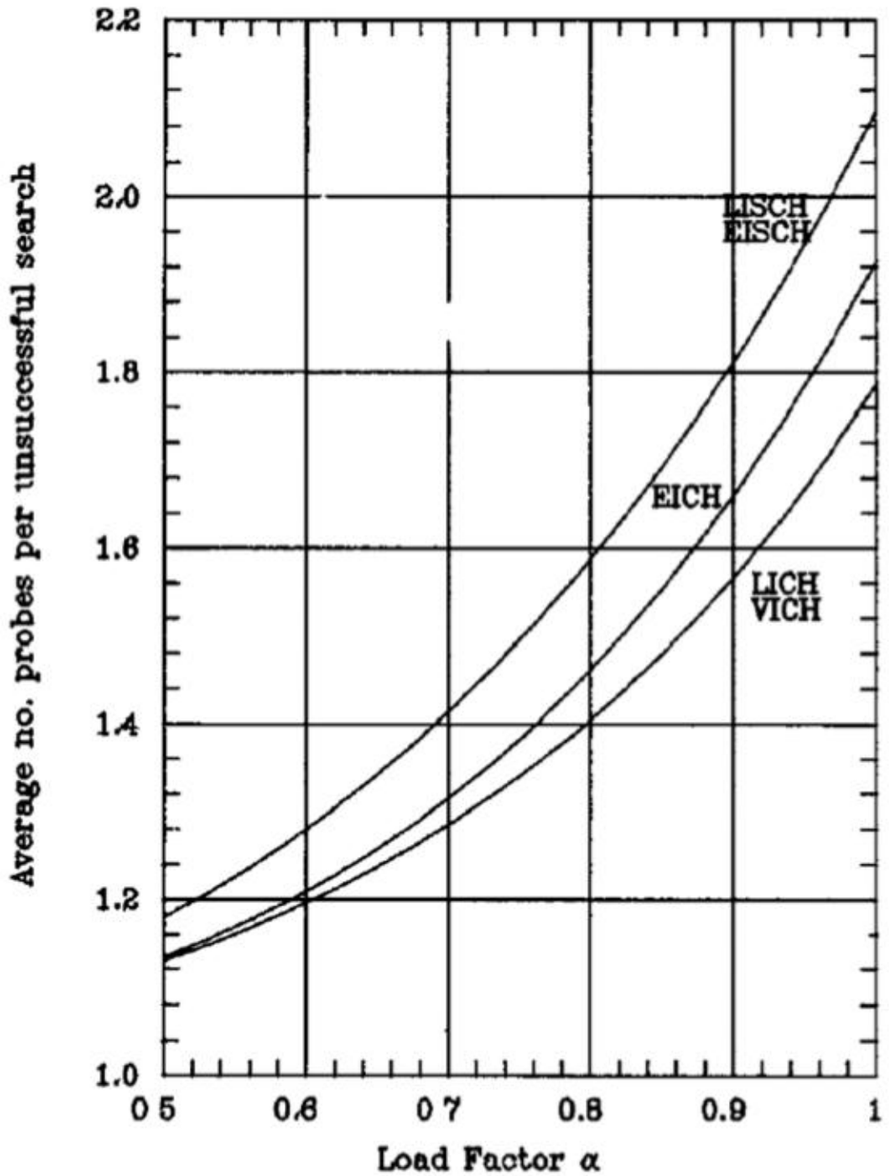
Efektivita srůstajícího hashování

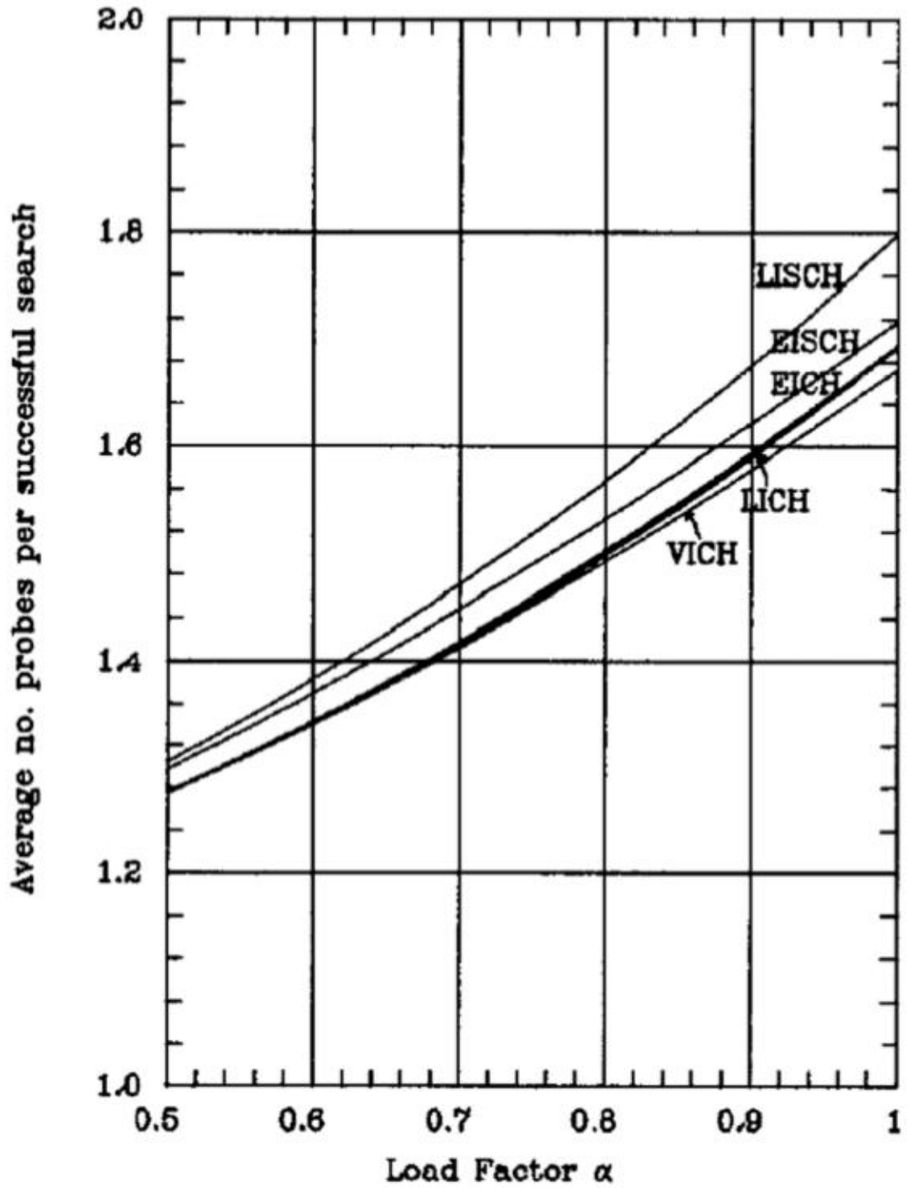
α method	0.2	0.4	0.6	0.8	0.9	0.95	0.99
EISCH	1.1065	1.2277	1.3684	1.5290	1.6182	1.6653	1.7033
LISCH	1.1063	1.2316	1.3789	1.5657	1.6737	1.7337	1.7827
BEISCH	1.1055	1.2286	1.3721	1.5336	1.6236	1.6728	1.7107
BLISCH	1.1055	1.2341	1.3836	1.5703	1.6818	1.7423	1.7898
REISCH	1.1063	1.2322	1.3693	1.5257	1.6124	1.6614	1.7014
RLISCH	1.1085	1.2384	1.3876	1.5653	1.6723	1.7296	1.7790
EICH	1.1116	1.2256	1.3408	1.4942	1.5867	1.6347	1.6762
LICH	1.1116	1.2256	1.3406	1.4888	1.5801	1.6281	1.6695

Source: Hsiao, Yeong-Shiou, and Alan L. Tharp, "Analysis of Other New Variants of Coalesced Hashing," Technical Report TR-87-2, Computer Science Department, North Carolina State University, 1987.

- Průměrný počet navštívených klíčů při operaci FIND. Vždy se předpokládá rovnoměrné rozložení klíčů po celém oboru hodnot hashovací funkce. Nemusí nutně odpovídat reálné situaci.

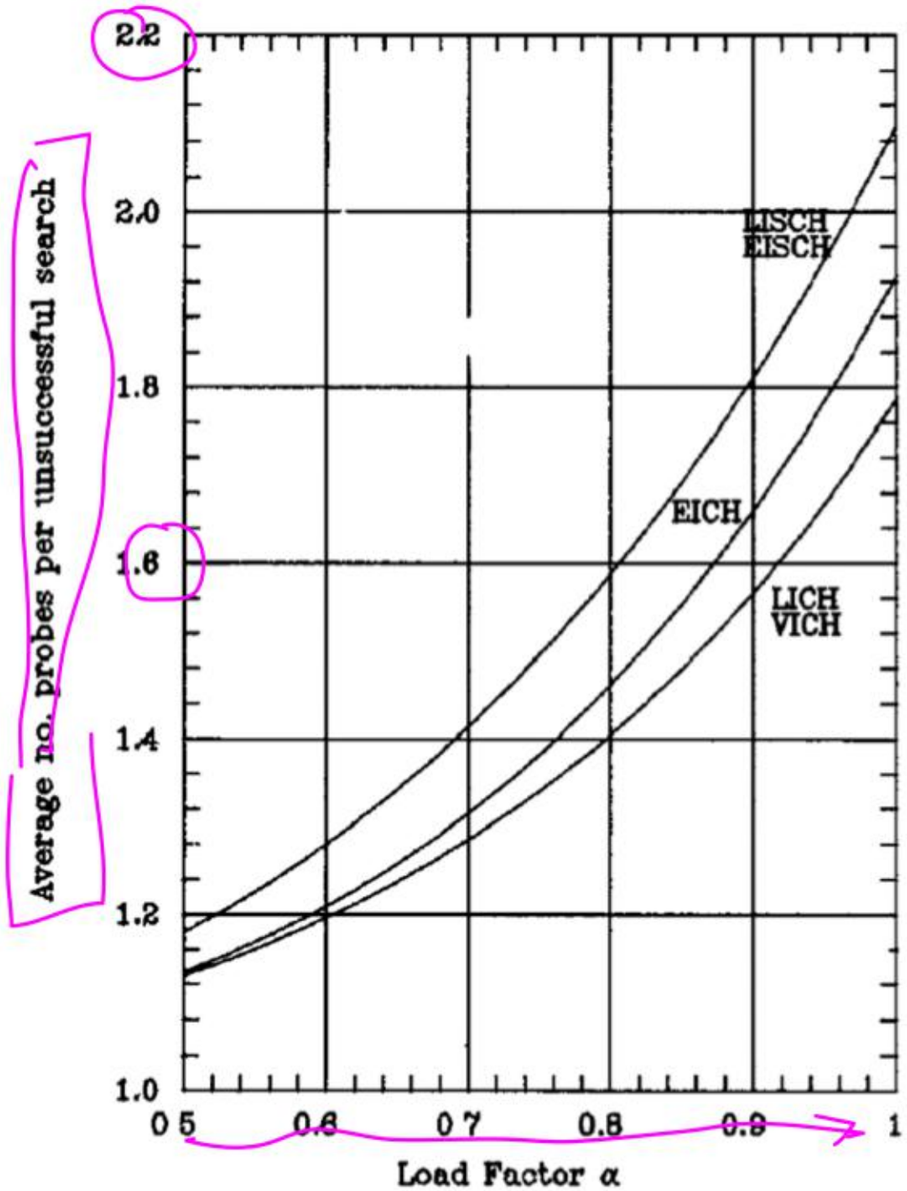
Celkové srovnání srůstajícího hashování

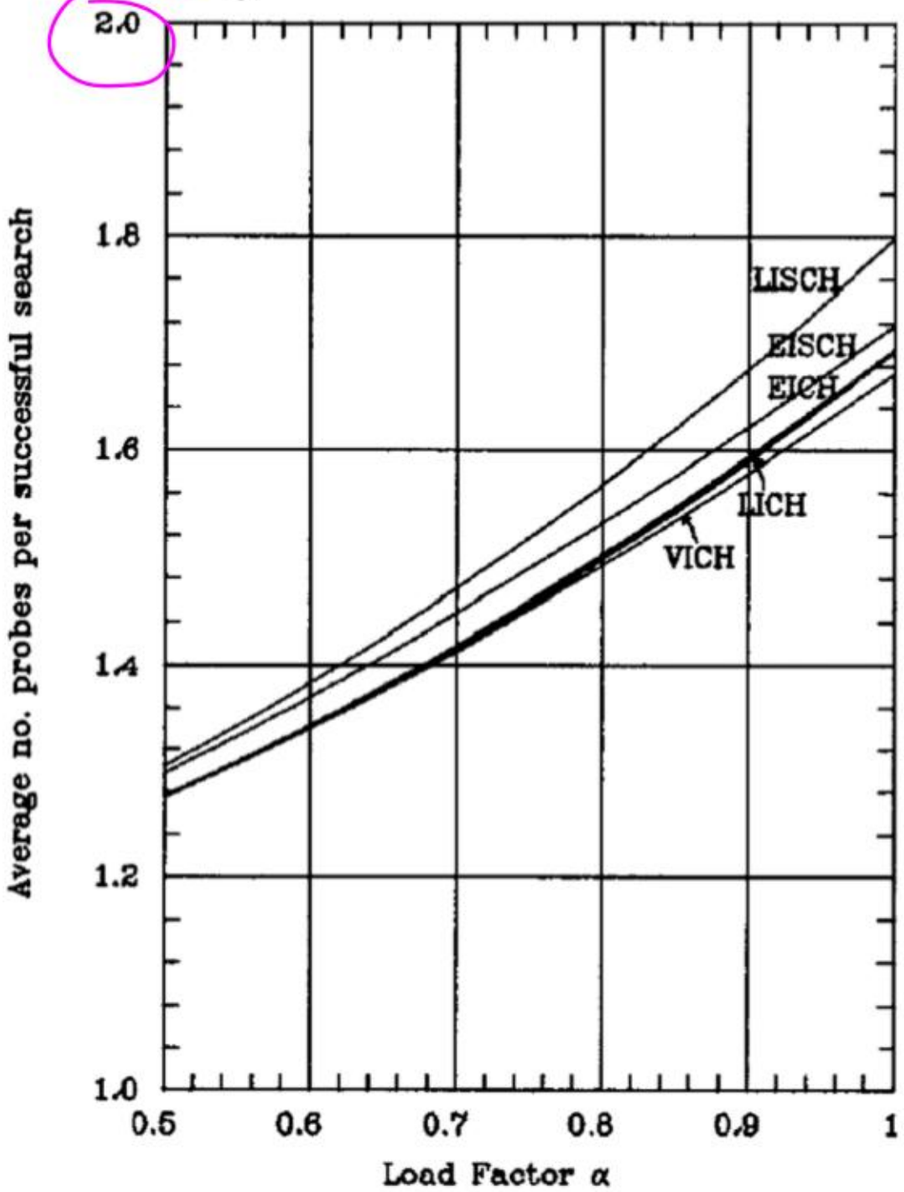




- S použitím sklepa vychází nejlépe VICH. Doporučená velikost β je 0,86.
- Bez sklepa vychází nejlépe EISCH.

Celkové srovnání srůstajícího hashování





- S použitím sklepa vychází nejlépe VICH. Doporučená velikost β je 0,86.
- Bez sklepa vychází nejlépe EISCH.

Efektivita srůstajícího hashování

α method	0.2	0.4	0.6	0.8	0.9	0.95	0.99
EISCH	1.1065	1.2277	1.3684	1.5290	1.6182	1.6653	1.7033
LISCH	1.1063	1.2316	1.3789	1.5657	1.6737	1.7337	1.7827
BEISCH	1.1055	1.2286	1.3721	1.5336	1.6236	1.6728	1.7107
BLISCH	1.1055	1.2341	1.3836	1.5703	1.6818	1.7423	1.7898
REISCH	1.1063	1.2322	1.3693	1.5257	1.6124	1.6614	1.7014
RLISCH	1.1085	1.2384	1.3876	1.5653	1.6723	1.7296	1.7790
EICH	1.1116	1.2256	1.3408	1.4942	1.5867	1.6347	1.6762
LICH	1.1116	1.2256	1.3406	1.4888	1.5801	1.6281	1.6695

Source: Hsiao, Yeong-Shiou, and Alan L. Tharp, "Analysis of Other New Variants of Coalesced Hashing," Technical Report TR-87-2, Computer Science Department, North Carolina State University, 1987.

- Průměrný počet navštívených klíčů při operaci FIND. Vždy se předpokládá rovnoměrné rozložení klíčů po celém oboru hodnot hashovací funkce. Nemusí nutně odpovídat reálné situaci.

Efektivita srůstajícího hash

α	0.2	0.4	0.6	0.8	0.9	0.99
method						
EISCH	1.1065					
LISCH	1.1063					
BEISCH	1.1055	1.2286				
[illegible]	1.1055	1.2341	1.3836	1.5703	1.6818	[illegible]
[illegible]	1.1063	1.2322	1.3693	1.5257	1.6124	[illegible]
RLISCH	1.1085	1.2384	1.3876	1.5653	1.6723	[illegible]
[illegible]	1.1116	1.2256	1.3408	1.4942	1.5867	[illegible]
LICH	1.1116	1.2256	1.3406	1.4888	1.5801	[illegible]

Source: Hsiao, Yeong-Shiou, and Alan L. Tharp, "Analysis of Other New Variants of Coalesced Hashing," Technical Report TR-87-2, Computer Science Department, North Carolina State University, 1987.

Průměrný poč navštívených míčů při operaci D. V y se předpokládá rovnoměrné rozložení klíčů po celém oboru hodnot hashovací funkce. Nemusí nutně odpovídat reálné situaci.