Úvod do databázových systémov

http://www.dcs.fmph.uniba.sk/~plachetk

/TEACHING/DB1

Tomáš Plachetka

Fakulta matematiky, fyziky a informatiky,
Univerzita Komenského, Bratislava

Zima 2026–2027

Cieľ a metodológia navrhovania databáz

Produktom logického návrhu relačnej databázy sú relácie, **atribúty a funkčné závislosti** (atribúty a funkčné závislosti sú dôležitejšie než relácie)

Na **overenie kvality návrhu**, resp. **automatické generovanie vhodnej organizácie dát** slúži **normalizácia** (formálne metódy)

Ľubovoľná relačná databáza sa dá reprezentovať jedinou tabuľkou. To však vedie k problémom:
redundancia, riziko nekonzistencie, anomálie pri vynechávaní a modifikácii dát, potreba NULL hodnôt, plytvanie pamäťou

Motivácia normalizácie (T. Conolly and C. Begg)

Príklad:

Staff Branch

staffNo	sName	position	salary	branchNo	bAddress
SL21	John White	Manager	30000	B005	22 Deer Rd, London
SG37	Ann Beech	Assistant	12000	B003	163 Main St, Glasgow
SG14	David Ford	Supervisor	18000	B003	163 Main St, Glasgow
SA9	Mary Howe	Assistant	9000	B007	16 Argyll St, Aberdeen
SG5	Susan Brand	Manager	24000	B003	163 Main St, Glasgow
SL41	Julie Lee	Assistant	9000	B005	22 Deer Rd, London

Staff

staffNo	sName	position	salary	branchNo
SL21	John White	Manager	30000	B005
SG37	Ann Beech	Assistant	12000	B003
SG14	David Ford	Supervisor	18000	B003
SA9	Mary Howe	Assistant	9000	B007
SG5	Susan Brand	Manager	24000	B003
SL41	Julie Lee	Assistant	9000	B005

Branch

branchNo	bAddress
B005	22 Deer Rd, London
B007	16 Argyll St, Aberdeen
B003	163 Main St, Glasgow

Funkčné závislosti (functional dependencies)

Definícia. V relácii r platí funkčná závislosť **X** $\rightarrow$ **Y** (t.j. množina atribútov **Y** funkčne závisí od množiny atribútov **X**), ak pre každú prípsutnú populáciu relácie r platí

$\forall X \, \forall Y_1 \, \forall Z_1 \, \forall Y_2 \, \forall Z_2 \, (r(X, Y_1, Z_1) \wedge r(X, Y_2, Z_2) \Rightarrow Y_1 = Y_2)$

kde X je inštancia **X**, Y_1 a Y_2 sú inštancie **Y**

Inými slovami, **X** $\rightarrow$ **Y** v relácii r hovorí, že ak sa v r ľubovoľné dva riadky zhodujú na množine atribútov **X**, tak potom sa zhodujú aj na množine atribútov **Y** (pre ľubovoľné prípustné naplnenie r)

Funkčné závislosti (functional dependencies)

Bar	Adresa	Pivo	Vyrobca	Cena
Janeway	Voyager	Bud	A.B.	3
Janeway	Voyager	WickedAle	Pete's	2
Spock	Enterprise	Bud	A.B.	3

Platí (vždy, nielen pre toto konkrétne naplnenie r):
Bar $\rightarrow$ Adresa
Bar, Pivo $\rightarrow$ Cena

Ale neplatí napríklad
Bar $\rightarrow$ Pivo
Pivo $\rightarrow$ Cena (hoci toto v tejto konkrétnej populácii databázy náhodou platí)

Vlastnosti funkčných závislostí (Armstrongove axiómy)

(A1) $\mathbf{X} \subseteq \mathbf{Y} \Rightarrow \mathbf{Y} \rightarrow \mathbf{X}$ reflexívnosť

(A2) $\forall \mathbf{Z}\ (\mathbf{X} \rightarrow \mathbf{Y} \Rightarrow \mathbf{XZ} \rightarrow \mathbf{YZ})$ rozšírenie (augmentation)

(A3) $(\mathbf{X} \rightarrow \mathbf{Y}) \wedge (\mathbf{Y} \rightarrow \mathbf{Z}) \Rightarrow \mathbf{X} \rightarrow \mathbf{Z}$ tranzitívnosť

Tvrdenia A1, A2, A3 sa v skutočnosti dajú dokázať z definície funkčnej závislosti v relačnom kalkule, napríklad:

(A1) $\forall \mathbf{X}\ \forall \mathbf{Y}\ \forall \mathbf{Z}_1\ \forall \mathbf{Z}_2\ ((\mathbf{X} \subseteq \mathbf{Y} \wedge r(\mathbf{Y}, \mathbf{Z}_1) \wedge r(\mathbf{Y}, \mathbf{Z}_2)) \Rightarrow \mathbf{X} = \mathbf{X})$

Prečo sa nazývajú „axiómy“? Lebo zakrátko dokážeme, že akékoľvek platné vlastnosti funkčných závislostí vieme vyjadriť len pomocou A1, A2, A3

Ďalšie vlastnosti funkčných závislostí

(B1) $(\mathbf{X} \rightarrow \mathbf{Y}) \wedge (\mathbf{X} \rightarrow \mathbf{Z}) \Rightarrow \mathbf{X} \rightarrow \mathbf{YZ}$ union rule

(B2) $(\mathbf{X} \rightarrow \mathbf{Y}) \wedge (\mathbf{WY} \rightarrow \mathbf{Z}) \Rightarrow \mathbf{WX} \rightarrow \mathbf{WZ}$ pseudotransivity

(B3) $(\mathbf{X} \rightarrow \mathbf{Y}) \wedge (\mathbf{Z} \subseteq \mathbf{Y}) \Rightarrow \mathbf{X} \rightarrow \mathbf{Z}$ decomposition

(B4) $(\mathbf{X} \rightarrow \mathbf{Y}) \wedge (\mathbf{X} \subseteq \mathbf{Z}) \Rightarrow \mathbf{Z} \rightarrow \mathbf{Y}$ left-hand side simplification

Dôkaz B1 (z Armstrongovych axióm):

$\mathbf{X} \rightarrow \mathbf{Y} \Rightarrow \mathbf{X} \rightarrow \mathbf{XY}$ podľa (A2)

$\mathbf{X} \rightarrow \mathbf{Z} \Rightarrow \mathbf{XY} \rightarrow \mathbf{YZ}$ podľa (A2)

Takže $(\mathbf{X} \rightarrow \mathbf{Y}) \wedge (\mathbf{X} \rightarrow \mathbf{Z}) \Rightarrow \mathbf{X} \rightarrow \mathbf{YZ}$ podľa (A3)

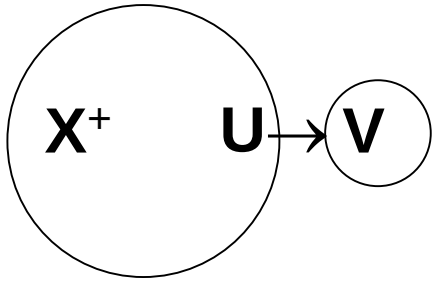
Dôkazy B2, B3 a B4 sa dajú urobiť podobným spôsobom s použitím Armstrongovych axióm

Uzáver množiny atribútov

Definícia. Nech **X** je množina atribútov a F množina funkčných závislostí. Potom uzáverom množiny atribútov **X** vzhľadom na F rozumieme množinu $\mathbf{X}^+$ všetkých atribútov Y takých, že $\mathbf{X} \rightarrow Y$ je logickým dôsledkom funkčných závislostí F

Výpočet uzáveru:

$\mathbf{X}^+ := \mathbf{X}$;
repeat
  for each $\mathbf{U} \rightarrow \mathbf{V} \in F$ **do**
    if $\mathbf{U} \in \mathbf{X}^+$ **then** $\mathbf{X}^+ := \mathbf{X}^+ \cup \mathbf{V}$;
while niečo sa pridalo do $\mathbf{X}^+$;



Optimalizácia: každá závislosť sa použije najviac raz, po použití ju možno vynechať. (Teda uzáver sa počíta v lineárnom čase.)

Výpočet uzáveru množiny atribútov

Príklad:

$AB \rightarrow C$ $\quad ACD \rightarrow B$ $\quad CG \rightarrow BD$

$C \rightarrow A$ $\quad D \rightarrow EG$ $\quad CE \rightarrow AG$

$BC \rightarrow D$ $\quad BE \rightarrow C$

Nech $\mathbf{X} = \{BD\}$. Potom $\mathbf{X}^+$ sa počíta takto:

$\mathbf{X}^{(0)} = \{B\mathbf{D}\}$,

$\mathbf{X}^{(1)} = \{\mathbf{B}D\mathbf{E}G\}$,

$\mathbf{X}^{(2)} = \{B\mathbf{C}D\mathbf{E}G\}$,

$\mathbf{X}^{(3)} = \{ABCDEG\}$,

$\mathbf{X}^{(4)} = \{ABCDEG\}$

$\mathbf{X}^{(3)} = \mathbf{X}^{(4)} = \mathbf{X}^+$

Úplnosť Armstrongovych axióm

Veta. Funkčná závislosť $\mathbf{X} \rightarrow \mathbf{Y}$ sa dá odvodiť z danej množiny funkčných závislostí F pomocou Armstrongovych axióm práve vtedy, keď $\mathbf{X} \rightarrow \mathbf{Y}$ je logickým dôsledkom F

Dôkaz:

$\Rightarrow$ Keďže Armstrongove axiómy sú pravdivé formuly, ktoré sa dajú dokázať z definície funkčnej závislosti, dajú sa z nich odvodiť len platné závislosti

$\Leftarrow$ Ostáva dokázať, že ak nejaká funkčná závislosť platí (t.j. je dôsledkom F), tak sa dá odvodiť z F len s použitím Armstrongovych axióm. Toto dokážeme sporom

Úplnosť Armstrongovych axióm

Predpokladajme, že **X** → **Y** platí a nedá sa odvodiť z F pomocou Armstrongovych axióm. Skonštruujme dvojriadkovú reláciu r:

	C	ostatné atribúty
r_0:	1 1 ... 1	0 0 ... 0
r_1:	1 1 … 1	1 1 ... 1

kde **C** je množina atribútov podobná $\mathbf{X}^+$ až na to, že tu uzáver robíme len s použitím Armstrongovych axióm, teda nie predošlým algoritmom (zatiaľ sme nedokázali, že **C** = $\mathbf{X}^+$, takže ich považujeme za rôzne). Platí $\mathbf{X} \in \mathbf{C}$ (podľa A1) a $\mathbf{Y} \notin \mathbf{C}$ (lebo predpokladáme, že **X** → **Y** sa nedá odvodiť). Teda **X** → **Y** nie je splnená v r (pre **X** sú v r rôzne **Y**).

Úplnosť Armstrongovych axióm

Teraz dokážeme, že v r je splnená ***každá*** funkčná závislosť, ktorá je dôsledkom F (spor s tvrdením, že $\mathbf{X} \rightarrow \mathbf{Y}$ nie je splnená v r).

Nech niektorá funkčná závislosť $\mathbf{S} \rightarrow \mathbf{T}$ je dôsledkom F, ale nie je splnená v r. Sú dve možnosti: 1.$\mathbf{S} \subseteq \mathbf{C}$ a 2.$\mathbf{S} \not\subseteq \mathbf{C}$.

1.Ak $\mathbf{S} \subseteq \mathbf{C}$, tak sa dá odvodiť $\mathbf{C} \rightarrow \mathbf{S}$ (podľa A1) a tiež $\mathbf{C} \rightarrow \mathbf{T}$ (podľa A3). Takže ak $\mathbf{S} \subseteq \mathbf{C}$, tak aj $\mathbf{T} \subseteq \mathbf{C}$ (**C** podľa definície obsahuje všetky odvoditeľné atribúty). Lenže v tom prípade závislosť $\mathbf{S} \rightarrow \mathbf{T}$ je splnená v r, lebo riadky r_0 a r_1 sa zhodujú na **T**.

2.Ak $\mathbf{S} \not\subseteq \mathbf{C}$, tak potom riadky r_0 a r_1 v r majú rôzne hodnoty na **S**, takže funkčná závislosť $\mathbf{S} \rightarrow \mathbf{T}$ je splnená triviálne (v takom prípade nezáleží na tom, či sa riadky r_0 a r_1 zhodujú na **T**).

Takže v r je splnená každá závislosť, ktorá je dôsledkom F. **QED**

Uzáver množiny funkčných závislostí

Definícia. Označme F^+ je množinu všetkých funkčných závislostí, ktoré sú dôsledkom funkčných závislostí z F (t.j. ktoré sa dajú odvodiť z F použitím Armstrongovych axióm). Množinu F^+ budeme nazývať uzáverom množiny funkčných závislostí F

Množina F^+ môže byť rozsiahla, t.j. **F^+ môže obsahovať exponenciálne veľa funkčných závislostí vzhľadom na počet atribútov a počet funkčných závislostí v F**

Pokrytie množiny funkčných závislostí

Definícia. Hovoríme, že množina funkčných závislostí G pokrýva množinu funkčných závislostí F, ak $G^+ \supseteq F^+$

Testovanie pokrytia priamo podľa tejto definície má exponenciálnu časovú zložitosť (lebo uzávery G^+, resp. F^+ môžu obsahovať exponenciálne veľa funkčných závislostí vzhľadom na |G| a |F|)

Stačí však testovať, či každú funkčnú závislosť z F možno odvodiť z G. **Testovanie pokrytia je teda polynomiálne** v čase

Príklad:

{AB $\rightarrow$ AC, B $\rightarrow$ A, C $\rightarrow$ B} pokrýva {AB $\rightarrow$ C, C $\rightarrow$ A}, ale nie naopak

Minimálne pokrytie množiny funkčných závislostí

Definícia. Funkčná závislosť sa nazýva kanonická, ak má na pravej strane práve jeden atribút

Definícia. Minimálne pokrytie množiny funkčných závislostí F je množina kanonických funkčných závislostí G taká, že G a F sa navzájom pokrývajú; a zároveň po vynechaní ľubovoľnej z funkčných závislostí z G alebo po vynechaní ľubovoľného atribútu na ľavej strane ľubovoľnej funkčnej závislosti z G prestane G pokrývať F

(Nejaké) minimálne pokrytie sa dá vypočítať v polynomiálnom čase vzhľadom na počet atribútov a vstupných funkčných závislostí

Minimálne pokrytie množiny funkčných závislostí

Príklad:

$AB \rightarrow C$, $D \rightarrow E$, $CG \rightarrow B$, $C \rightarrow A$, $D \rightarrow G$, $CG \rightarrow D$, $BC \rightarrow D$, $BE \rightarrow C$, $CE \rightarrow A$, $ACD \rightarrow B$, $CE \rightarrow G$

Minimálne pokrytia (vždy existuje aspoň jedno minimálne pokrytie, no môže ich existovať viac ako jedno):

$AB \rightarrow C$, $C \rightarrow A$, $BC \rightarrow D$, $D \rightarrow E$, $D \rightarrow G$, $BE \rightarrow C$, $CE \rightarrow G$, $CD \rightarrow B$, $CG \rightarrow D$

$AB \rightarrow C$, $C \rightarrow A$, $BC \rightarrow D$, $D \rightarrow E$, $D \rightarrow G$, $BE \rightarrow C$, $CE \rightarrow G$, $CG \rightarrow B$

Výpočet minimálneho pokrytia množiny funkč. závislostí

Algoritmus výpočtu minimálneho pokrytia množiny funkčných závislostí F:

1. Nahraď v F každú funkčnú závislosť $\mathbf{X} \rightarrow \mathbf{Y}$ množinou $\{\mathbf{X} \rightarrow A \mid A \in Y, A$ je jednoduchý atribút$\}$
2. Vynechaj postupne všetky redundantné atribúty na ľavých stranách $\mathbf{X} \rightarrow A$ (každý atribút z **X** treba testovať práve raz)
3. Vynechaj všetky redundantné závislosti $\mathbf{X} \rightarrow A$ (po vynechaní redundantnej funkčnej závislosti opakuj tento krok, kým žiadna funkčná závislosť nie je redundantná, každú závislosť treba testovať práve raz)

Krok 1 je triviálny

V kroku 2, pre každý atribút $B \in \mathbf{X}$ sa vypočíta uzáver $(\mathbf{X}\text{-}B)^+$ s použitím F. Ak $A \in (\mathbf{X}\text{-}B)^+$, tak odstráň B z **X**

V kroku 3 sa vypočíta uzáver $\mathbf{X}^+$, **ale len s použitím závislostí** F - $\{\mathbf{X} \rightarrow A\}$. Ak $A \in \mathbf{X}^+$, tak závislosť $\mathbf{X} \rightarrow A$ je redundantná

Výpočet minimálneho pokrytia množiny funkč. závislostí

F:

$AB \rightarrow C$ $ACD \rightarrow B$ $CG \rightarrow BD$ $C \rightarrow A$

$D \rightarrow EG$ $CE \rightarrow AG$ $BC \rightarrow D$ $BE \rightarrow C$

Krok 1:

$AB \rightarrow C$ $ACD \rightarrow B$ $CG \rightarrow B$ $CG \rightarrow D$

$C \rightarrow A$ $D \rightarrow E$ $D \rightarrow G$ $CE \rightarrow A$

$CE \rightarrow G$ $BC \rightarrow D$ $BE \rightarrow C$

Krok 2:

- Závislosť $ACD \rightarrow B$ sa nahradí $CD \rightarrow B$, lebo $B \in \{CD\}^+$ (keďže platí $C \rightarrow A$ a $ACD \rightarrow B$)
- Závislosť $CE \rightarrow A$ sa nahradí $C \rightarrow A$, lebo $A \in \{C\}^+$ (keďže platí $C \rightarrow A$)
- Žiadna ľavá strana sa už nedá skrátiť

Výpočet minimálneho pokrytia množiny funkč. závislostí

Krok 3:

$AB \rightarrow C$, $CD \rightarrow B$, $CG \rightarrow B$, $CG \rightarrow D$, $C \rightarrow A$, $D \rightarrow E$, $D \rightarrow G$, $C \rightarrow A$, $CE \rightarrow G$, $BC \rightarrow D$, $BE \rightarrow C$

- Závislosť **$C \rightarrow A$** je redundantná, lebo $C \rightarrow A$ je tam dvakrát
- Závislosť **$CG \rightarrow B$** je redundantná, lebo $CG \rightarrow D$, $CD \rightarrow B$, takže $B \in \{CG\}^+$, aj keď k výpočtu uzáveru nepoužijeme závislosť $CG \rightarrow B$

Minimálne pokrytie:

$AB \rightarrow C$	$CD \rightarrow B$	$CG \rightarrow D$
$C \rightarrow A$	$D \rightarrow E$	$D \rightarrow G$
$CE \rightarrow G$	$BC \rightarrow D$	$BE \rightarrow C$

(Iné minimálne pokrytie dostaneme, ak v kroku 3 vynecháme $CD \rightarrow B$, $CG \rightarrow D$, $C \rightarrow A$.)

Nadkľúče a kľúče

Definícia. Nech r je relácia nad množinou atribútov **U**. Potom množinu atribútov **K** takú, že **K** $\rightarrow$ **U**, nazývame nadkľúč relácie r (superkey, candidate key). Minimálny nadkľúč v zmysle množinovej inklúzie sa nazýva kľúč (key)

Príklad: Nech v r(A, B, C, D, E, F, G, H) platia funkčné závislosti

A $\rightarrow$ B, ABCD $\rightarrow$ E, EF $\rightarrow$ GH, ACDF $\rightarrow$ EG

Jediným kľúčom je ACDF, lebo ACDF nie sú na pravej strane žiadnej závislosti (takže musia patriť do každého kľúča) a všetky ostatné atribúty patria do uzáveru $\{ACDF\}^+$

Výpočet všetkých kľúčov

Treba prehľadať všetky množiny atribútov. Pre nájdenie všetkých kľúčov neexistuje lepší algoritmus ako exponenciálny v čase, lebo kľúčov môže byť exponenciálne veľa

Algoritmus zhora nadol:

Generuj lexikograficky zostupne (počínajúc celou množinou atribútov) **všetky podmnožiny atribútov**. Ak niektorá podmnožina nie je nadkľúčom, tak ju ďalej neredukuj. Ak niektorá podmnožina je nadkľúčom (v jej uzávere sú všetky atribúty), ale po odobratí ľubovoľného atribútu nadkľúčom nie je, tak je kľúčom

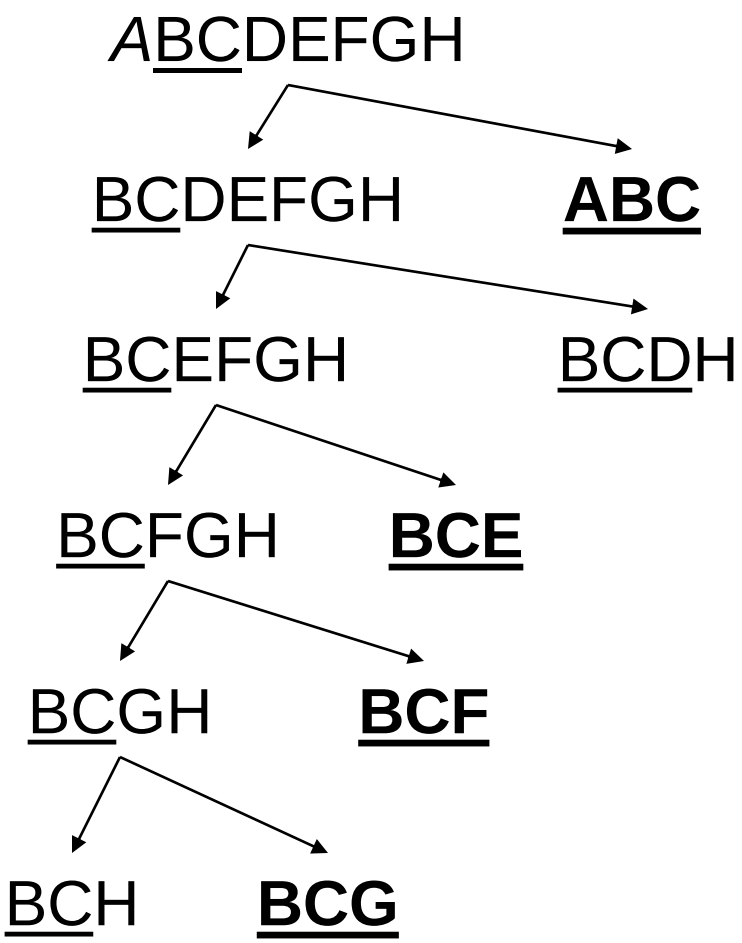
Praktický tip:

Atribúty, ktoré nie sú na pravej strane žiadnej funkčnej závislosti, musia byť v každom kľúči

Výpočet všetkých kľúčov zhora nadol: príklad

{G→F, F→A, AC→E, F→H, AH→F, E→G, H→D, BF→G}

Atribúty B a C musia byť v každom kľúči, lebo nie sú na žiadnej pravej strane



Prehľadávanie do hĺbky (backtrack):
V ľavej vetve sa odstráni 1 atribút, v pravej vetve je ten atribút v každej podmnožine (je podčiarknutý)

- Pred prehľadávaním stromu vypočítaj uzáver z množiny podčiarknutých atribútov a vynechaj nepodčiarknuté atribúty uzáveru—ak sú podčiarknuté atribúty nadkľúčom, tak to je kľúč (ukonči vetvu)
- V ľavej vetve over, či uzáver naďalej obsahuje ten atribút, ktorý sa v tej vetve vynecháva. Ak nie, ukonči vetvu
- Po nájdení nejakého kľúča neprehľadávaj podstromy, ktoré ten kľúč obsahujú

Dekompozícia relačnej schémy

Definícia. Množinu atribútov relácie r spolu s množinou funkčných závislostí, ktoré platia v r nazývame relačná schéma

Definícia. Dekompozícia relačnej schémy (r(**U**), F) je množina (r_1, F_1), ..., (r_n, F_n), kde každá z relácií r_1, ..., r_n je projekciou r na nejakej podmnožine atribútov r, pričom zjednotenie atribútov r_1, ..., r_n je **U**, a zároveň F pokrýva všetky F_1, ..., F_n (t.j. dekompozíciou nevznikajú žiadne nové funkčné závislosti)

Definícia. Dekompozícia (r_1, F_1), ..., (r_n, F_n) relačnej schémy (r, F) je bezstratová (spája sa bezstratovo), ak platí

$r = \Pi_{r_1}(r) \bowtie \Pi_{r_2}(r) \bowtie ... \bowtie \Pi_{r_n}(r)$ pre každú populáciu relácie r

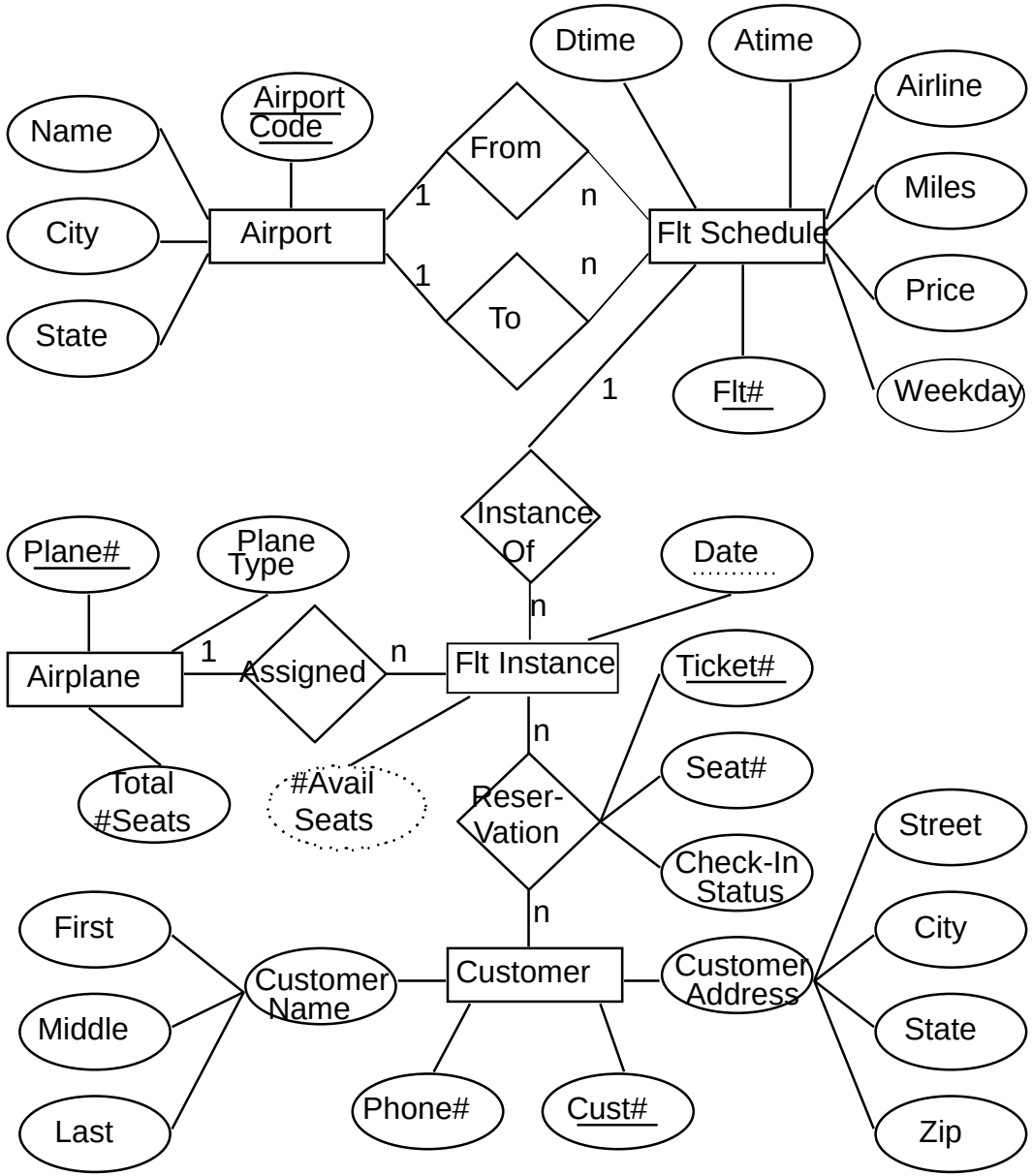
Normalizácia (Lee Mark)

The Four Commandments:

- **Thou Shalt Commit No Redundancy of Fact**
- **Thou Shalt Clutter No Facts**
- **Thou Shalt Preserve Information**
- **Thou Shalt Preserve Functional Dependencies**



Identifikácia funkčných závislostí (Lee Mark)



AIRPORT ↔ Airportcode

FLT-SCHEDULE ↔ Flt#

FLT-INSTANCE ↔ (Flt#, Date)

AIRPLANE ↔ Plane#

CUSTOMER ↔ Cust#

RESERVATION ↔ (Cust#, Flt#, Date)

RESERVATION ↔ Ticket#

Airportcode → name, City, State

Flt# → Airline, Dtime, Atime, Miles, Price, (from) Airportcode, (to) Airportcode

(Flt#, Date) → Flt#, Date, Plane#

(Cust#, Flt#, Date) →Cust#, Flt#, Date, Ticket#, Seat#, CheckInStatus,

Ticket# → Cust#, Flt#, Date

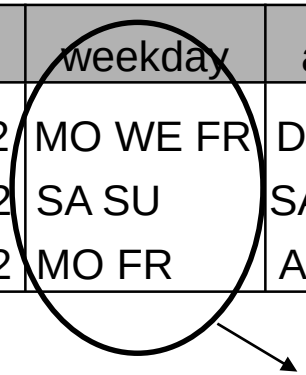
Cust# → CustomerName, CustomerAddress, Phone#

Normalizácia (Lee Mark)

BAD

FLT-SCHEDULE

flt#	weekday	airline	dtime	from	atime	to
DL242	MO WE FR	DELTA	10:40	ATL	12:30	BOS
SK912	SA SU	SAS	12:00	CPH	15:30	JFK
AA242	MO FR	AA	08:00	CHI	10:10	ATL



Attributes must be defined over domains with atomic values (1NF)

BETTER

FLT-SCHEDULE

flt#	weekday	airline	dtime	from	atime	to
DL242	MO	DELTA	10:40	ATL	12:30	BOS
DL242	WE	DELTA	10:40	ATL	12:30	BOS
DL242	FR	DELTA	10:40	ATL	12:30	BOS
SK912	SA	SAS	12:00	CPH	15:30	JFK
SK912	SU	SAS	12:00	CPH	15:30	JFK
AA242	MO	AA	08:00	CHI	10:10	ATL
AA242	FR	AA	08:00	CHI	10:10	ATL

Normalizácia (Lee Mark): Redundancy of Fact

BAD

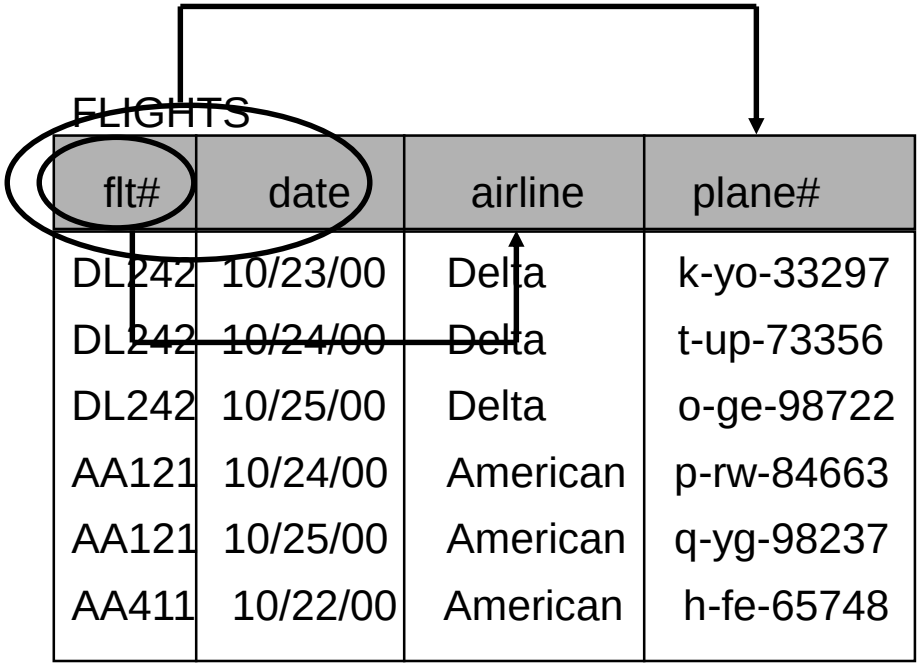
FLIGHTS

flt#	date	airline	plane#
DL242	10/23/00	Delta	k-yo-33297
DL242	10/24/00	Delta	t-up-73356
DL242	10/25/00	Delta	o-ge-98722
AA121	10/24/00	American	p-rw-84663
AA121	10/25/00	American	q-yg-98237
AA411	10/22/00	American	h-fe-65748

- **redundancy:** airline name repeated for same flight
- **inconsistency:** when airline name for a flight changes, it must (perhaps) be changed in many places

Normalizácia (Lee Mark): Fact Clutter

BAD



- **insertion anomalies:** how do we represent that SK912 is flown by Scandinavian without there being a date and a plane assigned?
- **deletion anomalies:** when we cancel AA411 on 10/22/00, we lose information that AA411 is flown by American (in other weeks)
- **update anomalies:** if DL242 is flown by Sabena, we must change it everywhere

Normalizácia (Lee Mark): Information Loss

BAD

DATE-AIRLINE-PLANE

date	airline	plane#
10/23/00	Delta	k-yo-33297
10/24/00	Delta	t-up-73356
10/25/00	Delta	o-ge-98722
10/24/00	American	p-rw-84663
10/25/00	American	q-yg-98237
10/22/00	American	h-fe-65748

FLIGHTS-AIRLINE

flt#	airline
DL242	Delta
AA121	American
AA411	American

Information loss (= false information): we polluted the database with false facts; we can't find the true facts

FLIGHTS, original

flt#	date	airline	plane#
DL242	10/23/00	Delta	k-yo-33297
DL242	10/24/00	Delta	t-up-73356
DL242	10/25/00	Delta	o-ge-98722
AA121	10/24/00	American	p-rw-84663
AA121	10/25/00	American	q-yg-98237
AA411	10/22/00	American	h-fe-65748

FLIGHTS, joined

flt#	date	airline	plane#
DL242	10/23/00	Delta	k-yo-33297
DL242	10/24/00	Delta	t-up-73356
DL242	10/25/00	Delta	o-ge-98722
AA121	10/24/00	American	p-rw-84663
AA121	10/25/00	American	q-yg-98237
AA121	*10/22/00*	*American*	*h-fe-65748*
AA411	*10/24/00*	*American*	*p-rw-84663*
AA411	*10/25/00*	*American*	*q-yg-98237*
AA411	10/22/00	American	h-fe-65748

Normalizácia (Lee Mark): Dependency Loss

BAD

FLIGHTS-AIRLINE

flt#	airline
DL242	Delta
AA121	American
AA411	American

DATE-AIRLINE-PLANE

date	airline	plane#
10/23/00	Delta	k-yo-33297
10/24/00	Delta	t-up-73356
10/25/00	Delta	o-ge-98722
10/24/00	American	p-rw-84663
10/25/00	American	q-yg-98237
10/22/00	American	h-fe-65748

- **dependency loss:** we lost the fact that (flt#, date) $\rightarrow$ plane#

Normalizácia (Lee Mark): Good Database Design

GOOD

FLIGHTS-AIRLINE

flt#	airline
DL242	Delta
AA121	American
AA411	American

FLIGHTS-DATE-PLANE

flt#	date	plane#
DL242	10/23/00	k-yo-33297
DL242	10/24/00	t-up-73356
DL242	10/25/00	o-ge-98722
AA121	10/24/00	p-rw-84663
AA121	10/25/00	q-yg-98237
AA411	10/22/00	h-fe-65748

- **no redundancy of *FACT (!)***
- **no inconsistency**
- **no insertion, deletion or update anomalies**
- **no information loss**
- **no dependency loss**

Redundancia: anomálie pri vkladaní

student

Snumber	Sname	Pnumber	Pname
s1	tamara	p1	tomas
s2	jozef	p2	jan

Ako pridáme profesora, ktorý (momentálne) nemá žiadnych študentov? Toto sa nedá bez použitia NULLs

Redundancia: anomálie pri vynechávaní

student

Snumber	Sname	Pnumber	Pname
s1	tamara	p1	tomas
s1	tamara	p2	jan
s2	jozef	p2	jan

Keď vynecháme študenta, máme zmazať riadok alebo nahradiť údaje o študentovi NULL hodnotami?

Ak zmažeme riadok, stratíme (občas) kompletnú informáciu o niektorom profesorovi

Ak nahradíme údaje o študentovi NULL hodnotami, tak niekedy vzniknú duplikáty (hoci v skutočnosti to nie sú doslova duplikáty, lebo hodnoty NULL sú navzájom neporovnateľné)

Redundancia: anomálie pri modifikácii

student

Snumber	Sname	Pnumber	Pname
s3	peter	p1	tamas
s1	tamara	p1	tamas
s2	jozef	p2	tamas
s2	jozef	p1	tamas

Ak zmeníme meno profesora (napríklad chceme odstrániť preklep a premenovať profesora tamas na tomas), tak musíme zmenu urobiť vo veľa riadkoch

A čo je horšie, **niekedy nesmieme tú zmenu urobiť**. Napríklad Tamas môže byť správne meno profesora s Pnumber=2, takže pri opravovaní preklepov v Pname sa musíme dívať tiež na pNumber

Úloha funkčných závislostí pri odstraňovaní redundancie

Uvažujme r(A, B, C)

• Ak v r neplatí žiadna funkčná závislosť, tak v r nie je žiadna redundancia vzhľadom na funkčné závislosti

• **Ak v r(A, B, C) platí A → B (ale neplatí A → C), tak niektoré riadky môžu mať rovnakú hodnotu A. Lenže potom budú mať aj rovnakú hodnotu B (ale rôzne hodnoty v C). Toto je problém**

• Tento problém sa dá odstrániť **dekompozíciou do r_1(A, B) a r_2(A, C). V r_1 už závislosť A → B nevadí**, keďže do r_1 nikdy nepridávame duplikát už existujúceho riadku. **V r_2 atribút B nie je**

Cieľom normalizácie je nájdenie „správnej“ dekompozície