Design Theory of Relational DBs.

- What makes a good database?
  - Normalization
  - Avoid "anomalies"

Functional Dependencies (FD)

- A FD is a constraint between two set of attributes of a relation.
- Given R, a set of attributes X in R is said to functionally determine another set of attributes Y in R ($X \rightarrow Y$) iff 2 tuples have the same values of attributes X then they must have the same values for attributes Y.

We write them as:

$$\underbrace{A_1 \ldots A_n}_{} \rightarrow \underbrace{B_1 \ldots B_m}_{}$$

Attributes written as list.

Example:

title	Year	Lenght	Genre	studioName	starName
Star Wars	1977	124	SciFi	Fox	Carry Fisher
Star Wars	1977	124	SciFi	Fox	Mark Hamill
Star Wars	1977	124	SciFi	Fox	Harrison Ford
The Godfather	1972	175	Drama	Paramount	Robert Duvall
The Godfather	1972	175	Drama	Paramount	Marlon Brandon
Apocalypse Now	1953	153	War	Zoetrope	Marlon Brandon

I claim by design that

title, year $\rightarrow$ lenght, genre, studioName

title, year $\not\rightarrow$ starName

Superkey (SK)

A set of attributes $\{A_1 \ldots A_n\}$ is a superkey of R iff $A_1 \ldots A_n \rightarrow R$

Candidate key (key)

A candidate key is a superkey that is minimal:

There is no proper subset of C of $\{A_1 \ldots A_n\}$ s.t $C \rightarrow R$

One candidate key becomes the Primary key!

For our example:

- All attributes of R are always a SK.
- title, year, starname is a candidate key

  title, year, starName $\rightarrow$ R.

- Any superset of a candidate key is a SK.

Reasoning about FD's.

- Given a relation R two sets of FDs A & B are equivalent if:

  The set of instances of R that satisfy A is exactly the same that satisfy B

- A follows from B if every instance of R that satisfies B also satisfies A.
- A & B are equivalent iff:

  A follows B and B follows A

Armstrong's Axioms (3.2 page 81)

Given relation R with subsets of attributes
$X, Y, Z \subseteq R$

Reflexivity (Trivial)
$Y \subseteq X$ then $X \rightarrow Y$

Augmentation:
If $X \rightarrow Y$ then $XZ \rightarrow YZ$ for any Z

Transitivity:
If $X \rightarrow Y$, $Y \rightarrow Z$ then $X \rightarrow Z$

Additional Rules.

They can be derived from axioms:

Union:

If $X \rightarrow Y$, $X \rightarrow Z$ then $X \rightarrow YZ$

Decomposition

If $X \rightarrow YZ$ then $X \rightarrow Y$ and $X \rightarrow Z$

Ex :

Derive Union from axioms.

$X \to Y$, $X \to Z$

$XZ \to YZ$ Augmentation

$XX \to XZ$
$X \to XZ$ } Augmentation.

$X \to XZ \to YZ$
$X \to YZ$ ∎ } Transitivity

Derive decomposition

$X \to YZ$

$YZ \to Y$
$YZ \to Z$ } Reflexivity.

$X \to Y$
$X \to Z$ ∎ } Transitivity.

Closure of attributes (3.2.4)

Given a relation R and a set f of FDs, what other FDs can be computed from a set of FDs f?

The closure of a set of attributes $A_1 \ldots A_n$ denoted $\{A_1 \ldots A_n\}^+$ is the set of attr. that can be derived from $A_1 \ldots A_n$ using f.

Hard to do and error prone via axioms!!

Alg:

1) Rewrite FDs in f in canonical form
2) $X \leftarrow A_1 \ldots A_n$
3) for each $B_1 \ldots B_m \rightarrow C$ in FDs
   if $B_1 \ldots B_m \subseteq X$ and $C \notin X$
   add C to X
4) Repeat (3) until X does not change

X is $\{A_1 \ldots A_n\}^+$

Ex:

$R(A\,B\,C\,D\,E)$

$$f = \begin{cases} AB \to C \\ BC \to AD \\ D \to E \\ CF \to B \end{cases}$$

①
$AB \to C$ 1
$BC \to A$ 2
$BC \to D$ 3
$D \to E$ 4
$CF \to B$ 5

Compute $\{AB\}^+$

$X \Leftarrow AB$

First pass:

$X \Leftarrow ABC$ fd 1

$X \Leftarrow ABCD$ fd 3

$X \Leftarrow \underline{\underline{ABCDE}}$ fd 4

all attributes hence X will not change any more

$\{AB\}^+ = \{ABCDE\}$

AB is a SK of R. Is it a candidate key?

Closure of attr. can help us find CKs of a relation: Compute $\{A\}^+$ and $\{B\}^+$

Closure of set of FDs

Given a set f of FDs, its closure f^+ is the set of all FDs derived from f.

Ex:

$$f = \left\{ \begin{array}{l} A \to B \\ B \to C \end{array} \right\} \qquad A \to C \in f^+$$

Two sets A & B of FDs are equivalent iff. $\{A\}^+ = \{B\}^+$

$$\left\{ \begin{array}{l} A \to B \\ B \to C \end{array} \right\}^+ = \left\{ \begin{array}{l} A \to B \\ B \to C \\ A \to C \end{array} \right\}^+$$

We can easily test if $X \to Y \in f^+$

$X \to Y \in f^+$ iff $Y \subseteq \{X\}^+$ using f.

Ex:

$$A \to C \in \left\{ \begin{array}{l} A \to B \\ B \to C \end{array} \right\}$$

$$\{A\}^+ = \{ABC\} \Rightarrow A \to ABC$$

$$\Rightarrow \begin{array}{l} A \to A \\ A \to B \\ A \to C \end{array}$$

Basis of a relation

Given a relation R and FDs f we say that any set g s.t. $f^+ = g^+$ is a **basis** of R.

Minimal Basis of FDs. (3.2.7)

Any relation R has many equivalent set of FDs (many basis-es).

To avoid an explosion of FDs we usually use a minimal basis

A **minimal basis** B of a relation R is a basis of R s.t.

1) All FDs in B are in canonical form

2) If for any FD we remove one or more attr. from the left hand side the result is no longer a basis,

3) If any FD is removed from B, the result is no longer a basis

Ex: Is A a minimal basis of B?

$$A = \left\{ \begin{array}{l} A \to B \\ B \to A \\ B \to C \\ C \to B \end{array} \right\} \qquad B = \left\{ \begin{array}{ll} A \to B & (1) \\ A \to C & (2) \\ B \to A & (3) \\ B \to C & (4) \\ C \to A & (5) \\ C \to B & (6) \\ AB \to C & (7) \\ AC \to B & (8) \\ BC \to A & (9) \end{array} \right\}$$

Given FDs in A, can we generate FDs in B?

(1), (3), (4), (6) already in A.

Can we generate (2), (5), (7), (8), (9)?

$\{A\}^+ = \{A\,B\,C\}$.

$\Rightarrow$ (2) can be generated, and also (7), (8) by using augmentation. (or compute $\{AB\}^+$, $\{AC\}^+$)

(5)? $\{C\}^+ = \{C\,B\,A\}$ $\Rightarrow$ yes (5) and (9) (augmentation) can be generated.

So from A we can generate B.

Hence A is a basis of B

B is also a basis of A $(A \subset B)$

Is A minimal?

- Can we drop $A \to B$?

$A \to B$ be generated from $\left\{ \begin{array}{l} B \to A \\ C \to B \\ B \to C \end{array} \right\}$

$\{A\}^+ = \{A\}$ so no, $A \to B$ cannot be removed.

- Can we drop $B \to A$?

$B \to A$ be generated from $\left\{ \begin{array}{l} A \to B \\ C \to B \\ B \to C \end{array} \right\}$

$\{B\}^+ = \{B\ C\}$, so no, it can not be removed.

- Repeat for $C \to B$ and $B \to C$.

Yes, it is minimal.

Another Example.

Given
$$\left\{\begin{array}{l} AC \to D \\ AD \to C \\ A \to CD \\ C \to D \end{array}\right.$$
Compute its minimal cover

1) Write in canonical form

① $AC \to D$ ③ $A \to C$ ⑤ $C \to D$

② $AD \to C$ ④ $A \to D$

2) Remove redundant attr. for LHS

Test A in ①

Can we generate A from C?

$\{C\}^+ = CD$ <u>NO</u>

Test C:

Can we generate C from A:

$\{A\}^+ = AC\ldots$ <u>yes</u>

Drop C from ① $\Rightarrow$ <u>$A \to D$</u>. (Replace ①)

---

②

Can we generate A from D? <u>NO</u>

Can we generate D from A? Yes.

$\Rightarrow$ $A \to C$ (Replace ②)

Now we have.

1. $A \rightarrow D$
2. $A \rightarrow C$
3. $A \rightarrow C$
4. $A \rightarrow D$
5. $C \rightarrow D$

③ Remove redundant FDs.

① and ② are obviously redundant.

$\Rightarrow$ Remove.

New FDs:

③ $A \rightarrow C$

④ $A \rightarrow D$

⑤ $C \rightarrow D$

Can ③ be generated from ④, ⑤?

No. <u>Keep.</u>

Can ④ be generated from ③ ⑤?

Yes. Remove.

Can ⑤ be generated from ③?

No. Keep.

<u>Minimal</u> <u>Cover</u>

③ $A \rightarrow C$

⑤ $C \rightarrow D$

Projection of FDs (3.2.8)

Given R and set F of FDs the projection of F on $R_1 = \pi_L R$ is the set of FDs that follows from F that involve only attributes in R_1.

Algorithm:

$T \leftarrow \emptyset$

for each subset $X \in L$ compute $\{X\}^+$

for every attribute A in $\{X\}^+$

add $X \rightarrow A$ to T

iff $A \in L$ and

$A \notin X$ (nontrivial)

Ex: $R(ABCD)$ $F = \left\{ \begin{array}{l} A \to B \\ B \to C \\ C \to D \end{array} \right\}$

Compute FDs of $\pi_{ACD} R$

1)	② closure
A C D	$\cancel{A}\ \cancel{B}\ \cancel{C}\ \cancel{D}$
A C	$\cancel{A}\ \cancel{C}\ \cancel{B}\ D$
A D	$\cancel{A}\ \cancel{D}\ \cancel{B}\ C$
A	$\cancel{A}\ \cancel{B}\ C\ D$
C D	$\cancel{C}\ \cancel{D}$
C	$\cancel{C}\ D$
D	$\cancel{D}$

3) Remove RHS att not in L.

4) Remove trivial FDs.

Result: $\left\{ \begin{array}{ll} AC \to D & A \to C \\ AD \to C & A \to D \\ & C \to D \end{array} \right\}$

Is it a minimal basis? <u>No</u>

$\left\{ \begin{array}{l} A \to D \\ AC \to D \\ AD \to C \end{array} \right\}$ can be generated from $\left\{ \begin{array}{l} A \to C \\ C \to D \end{array} \right\}$

Prove it!!

Design of Relational DBs (3.3)

title	Year	Lenght	Genre	studioName	starName
Star Wars	977	124	Sci Fi	Fox	Carry Fisher
Star Wars	977	124	Sci Fi	Fox	Mark Hamill
Star Wars	977	124	Sci Fi	Fox	Harrison Ford
The Godfather	972	175	Drama	Paramount	Robert Duvall
The Godfather	972	175	Drama	Paramount	Marlon Brandon
Apocalypte Now	1953	153	War	Zoetrope	Marlon Brandon

FDs: title, year → length, studio Name

Anomalies

- Redundancy: Unnecessary repeated info
- Update anomalies: If we change one tuple we might have to change another: Ex: change Lenght of a movie.
- Deletion anomalies: If we delete a tuple we might lose other info: Ex: Remove M. Brandon from Ap. Now.

Decomposing Relations

To deal with anomalies we decompose relations.

Given $R(A_1 .. A_n)$ a decomposition into $S(B_1 .. B_m)$ and $T(C_1 .. C_k)$ s.t.

1) $\{A_1 \dots A_n\} = \{B_1 .. B_m\} \cup \{C_1 \dots C_k\}$

and

2) $S = \Pi_{B_1 \dots B_m} R$ and

$T = \Pi_{C_1 \dots C_k} R$

We can decompose Movies into

S(title, year, length, genre, studioName)

T(title, year, starName)

title	Year	Lenght	Genre	studioName
Star Wars	1977	124	Sci Fi	Fox
The Godfather	1972	175	Drama	Paramount
Apocalyte Now	1953	153	War	Zoetrope

= S

Call this Movies2

title	year	starName
Star Wars	1977	Carry Fisher
Star Wars	1977	Mark Hamill
Star Wars	1977	Harrison Ford
The Godfather	1972	Robert Duvall
The Godfather	1972	Marlon Brandon
Apocalypse Now	1953	Marlon Brandon

$= T.$

Call this Movies3

Good decompositions.

Given a relation R we want to decompose it into two relations S and T s.t.

1) $R = S \bowtie T$ lossless join

2) The projection of FDs F_R of R into S (F_S) and T (F_T) satisfies:

$$\{F_S \cup F_T\}^+ = \{F_R\}^+$$

Dependency preserving.

Boyce Codd Normal Form (BCNF)

A relation R is in BCNF iff for every non trivial FD

$A_1 \ldots A_n \rightarrow B_1 \ldots B_m$

$A_1 \ldots A_n$ is a Superkey.

Ex:

Movies is not BCNF

title, year → lenght, studioName

is a not a SK of Movies

If a relation R is not BCNF then decompose into relations $R_1 \ldots R_n$

s.t $R_1 \bowtie R_2 \ldots \bowtie R_n = R$

⇒ Loss-less join decomposition.

Algorithm to decompose into BCNF relations

Given R and set F of FDs:

R is <u>not</u> BCNF.

1) Choose one FD $X \rightarrow Y$ not in BCNF

2). Decompose:

$$R \xrightarrow{X \rightarrow Y} R_1 = \{X\}^+, \quad R_2 = X \cup (R - R_1)$$

3) Compute FDs for R_1 and R_2 (projection of FDs of R into R_1, R_2)

4) If R_1 or R_2 are not BCNF recursively decompose.

Guaranteed to be lossless join. but not FD preserving.

Any non BCNF relation has a BCNF loss-less join decomposition but not a BCNF FD preserving decomposition.

Ex: $R(TCH)$

$H \to C$
$TC \to H$.

$H \to C$ not BCNF.

TCH.

$H \to C$

$\{H\}^+ = HC$ ← any 2 att rel is BCNF ↗ $H \cup \{TCH - HC\} = HT$

FDs

~~HC~~	~~HC~~
H	$\cancel{H}C$
~~C~~	~~C~~

$H \to C$

FDs

~~HT~~	~~$HT\cancel{C}$~~
~~H~~	~~$H\cancel{C}$~~
~~T~~	~~T~~

$\emptyset$

Decomposition $R_1 = HC$ $FD_1 = \{H \to C\}$

$R = HT$ $FD_2 = \emptyset$

Not FD preserving lost $TC \to H$.

3rd Normal Form (3NF)

If we cannot decompose a relation into BCNF relations that are FD preserving we are happy if they can be decompose into 3NF relations.

Any relation R with a set of FDs F has a 3NF decomposition that is loss-less join and FD preserving.

A relation R with FDs F is in 3NF if for every non trivial FD

$$A_1 \ldots A_n \rightarrow B_1 \ldots B_m$$

- it is a SK

or

- $C \in \{B_1 \ldots B_m\}$ is either

  $C \in \{A_1 \ldots A_n\}$ or

  C is part of <u>some</u> candidate key.

Ex: $R(ABCDE)$

$$F = \left\{ \begin{array}{l} AB \to C \\ C \to B \\ A \to D \end{array} \right\}$$

Is it 3NF?

$AB \to C$ AB not a SK.

is C part of a CK?

Need to compute candidate keys of R

- Heuristic:

  AE never on righthand side of FD

  $\Rightarrow$ always part of a key.

Use closure of attr. to compute SKs:

all combination of attr. $\Downarrow$ closure.

all combination of attr.	closure
AE BCD	AEBCD
AE BC	AEBCD
AE B D	AEBDC
~~AE B~~	AEBCD
AE CD	AECDB
~~AE C~~	AECBD
AE D	AED
AE	AED.

all SK.

minimal $\Rightarrow$ Candidate keys. $\left\{ \begin{array}{l} AEB \\ AEC \end{array} \right.$

Back to testing if R is 3NF.

$AB \to C$ C is part of CK. ✓

$C \to B$ C is not SK. but
B is part of CK.

$A \to D$ A is not SK.
D is not part of CK

$\Rightarrow$ R is not 3NF. $\blacksquare$

Decomposition of a Relation into 3NF relations that is loss-less join and FD preserving: (Synthesis alg 3.5.2)

Given R with set of FDs F

1) Find G, a minimal basis of F
2) For each FD $A_1 \ldots A_n \to B$ in G.
   add a relation with schema
   $A_1 \ldots A_n B$ with FD
   $A_1 \ldots A_n \to B$
3) If none of the added relations in step 2 is a SK of R add another relation whose schema is a key of R.

Ex:

$R(ABCDE)$

$F = \{ AB \to C, \; C \to B, \; A \to D \}$ F is minimal basis

$R_1 = ABC$ with $AB \to C$

$R_2 = CB$ with $C \to B$

$R_3 = AD$ with $A \to D$

None of them contains a SK.
(see previous exercise, you can verify it by computing $\{R_i\}^+$)
We know its keys are AEB and AEC
We need only one. So add

$R_4 = AEB$ with no FDs.

Decomposition of R is
R_1, R_2, R_3 and R_4 with corresponding FDs