Pertemuan 9

Functional Dependencies

tyas@stikom.edu

Chapter Outline

- 1 Panduan Desain Informal untuk Relational Databases
  - 1.1Semantics of the Relation Attributes
  - 1.2 Redundant Information in Tuples and Update Anomalies
  - 1.3 Null Values in Tuples
  - 1.4 Spurious Tuples
- 2 Functional Dependencies (FDs)
  - 2.1 Definition of FD
  - 2.2 Inference Rules for FDs
  - 2.3 Equivalence of Sets of FDs
  - 2.4 Minimal Sets of FDs

tyas@stikom.edu

1 Panduan Desain Informal untuk Relational Databases(1)

- Apa yang dimasud dengan ***Relational Database Design***?
  - Pengelompokkan atribut menjadi schema relasi yang "baik"
  - Dua level dari schema relasi
    - The logical "user view" level
    - The storage "base relation" level
- Design umunya berhubungan dengan base relations
- Apa yang menjadi kriteria untuk relasi dasar yang "baik"?

tyas@stikom.edu

Panduan Desain Informal untuk Relational Databases(2)

- Pertama akan membahas tentang panduan informal untuk desain relasi yang baik
- Kemudian tentang konsep dari *Functional Dependencies* dan *Normal Forms*
  - - 1NF (First Normal Form)
  - - 2NF (Second Normal Form)
  - - 3NF (Third Normal Form)
  - - BCNF (Boyce-Codd Normal Form)

tyas@stikom.edu

1.1 Semantics of the Relation Attributes

- Panduan 1: secara Informal, setiap *tuple* dalam relasi seharusnya mewakili satu *entity* atau *relationship instance*.
  - Atribut-atribut dari entity yang berbeda (EMPLOYEEs, DEPARTMENTs, PROJECTs) seharusnya tidak tercampur dalam relasi yang sama
  - Hanya *Foreign Keys* yang seharusnya menghubungkan dengan entity yang lain
  - *Entity* dan *relationship attributes* seharusnya terpisah sebanyak mungkin
- Intinya: Desain sebuah schema yang dapat menjelaskan secara mudah tiap relasinya. S*emantics* dari setiap attributes seharusnya mudah untuk dimengerti.

tyas@stikom.edu

Figure 10.1 A simplified COMPANY relational database schema

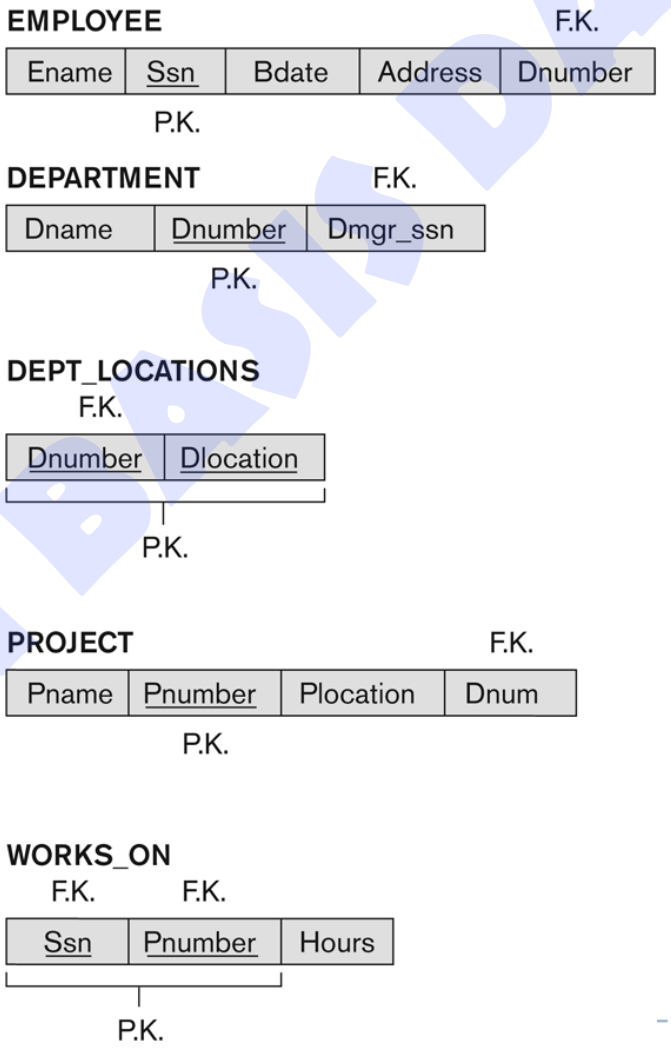


Figure 10.1
A simplified COMPANY relational database schema.

tyas@stikom.edu

1.2 Redundant Information in Tuples and Update Anomalies

- Informasi yang disimpan secara redundant
  - Membuang banyak tempat penyimpanan
  - Menyebabkan masalah dengan *update anomalies*
    - Insertion anomalies
    - Deletion anomalies
    - Modification anomalies

tyas@stikom.edu

EXAMPLE OF AN UPDATE ANOMALY

- Consider the relation:
  - EMP_PROJ(Emp#, Proj#, Ename, Pname, No_hours)
- Update Anomaly:
  - Mengganti nama dari *project number P1* dari *"Billing"* menjadi *"Customer-Accounting"* mungkin akan menyebabkan harus mengganti semua 100 karyawan yang bekerja pada project P1.

tyas@stikom.edu

EXAMPLE OF AN INSERT ANOMALY

- Consider the relation:
  - EMP_PROJ(Emp#, Proj#, Ename, Pname, No_hours)
- Insert Anomaly:
  - Tidak bisa menambahkan sebuah project kecuali ada pegawai yang ditugaskan disana.
- Sebaliknya
  - Tidak bisa menambahkan pegawai kecuali ditugaskan dalam sebuah project.

tyas@stikom.edu

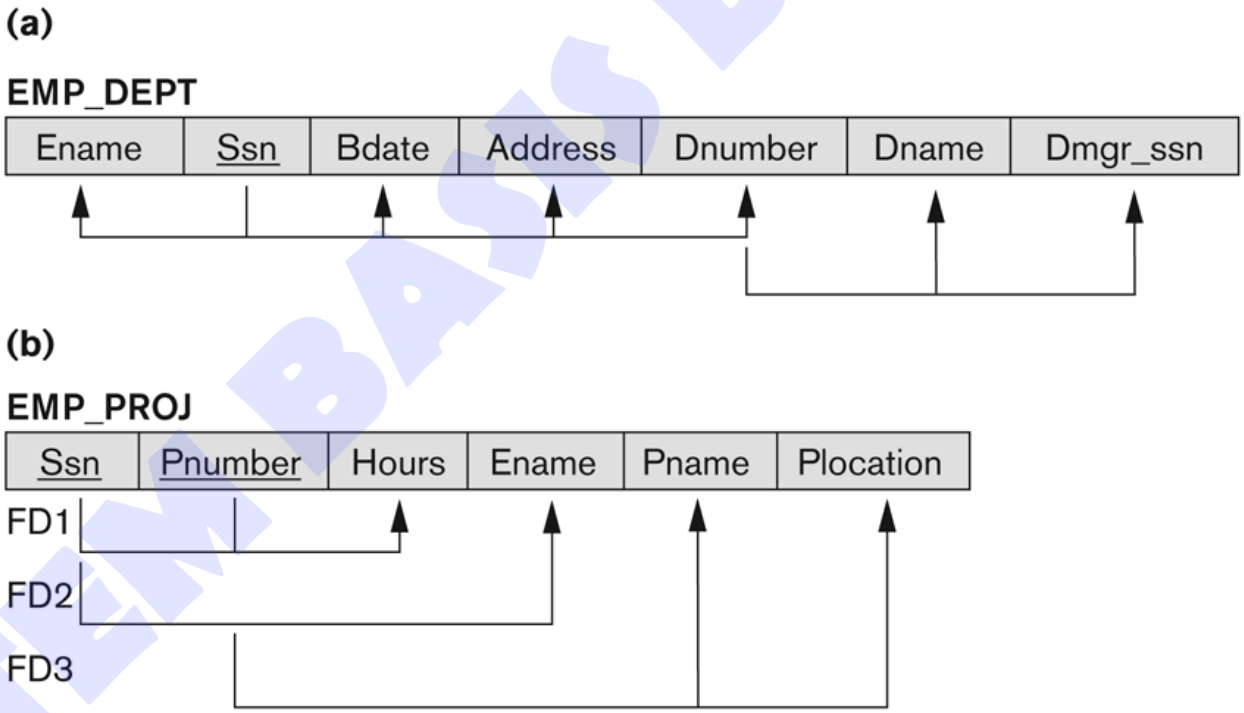
EXAMPLE OF AN DELETE ANOMALY

- Consider the relation:
  - EMP_PROJ(Emp#, Proj#, Ename, Pname, No_hours)
- Delete Anomaly:
  - Ketika sebuah project dihapus, akan mengakibatkan penghapusan semua pegawai yang bekerja pada project tersebut.
  - secara bergantian, jika seorang pegawai hanya bekerja sendirian pada suatu project, maka penghapusan pegawai itu akan berakbat dengan penghapusan project tersebut.

tyas@stikom.edu

Figure 10.3 Two relation schemas suffering from update anomalies

Figure 10.3
Two relation schemas suffering from update anomalies.
(a) EMP_DEPT and
(b) EMP_PROJ.



tyas@stikom.edu

Figure 10.4 Example States for EMP_DEPT and EMP_PROJ

Figure 10.4
Example states for EMP_DEPT and EMP_PROJ resulting from applying NATURAL JOIN to the relations in Figure 10.2. These may be stored as base relations for performance reasons.

Redundancy

EMP_DEPT

Ename	Ssn	Bdate	Address	Dnumber	Dname	Dmgr_ssn
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291 Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975 FireOak, Humble, TX	5	Research	333445555
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5	Research	333445555
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4	Administration	987654321
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1	Headquarters	888665555

Redundancy Redundancy

EMP_PROJ

Ssn	Pnumber	Hours	Ename	Pname	Plocation
123456789	1	32.5	Smith, John B.	ProductX	Bellaire
123456789	2	7.5	Smith, John B.	ProductY	Sugarland
666884444	3	40.0	Narayan, Ramesh K.	ProductZ	Houston
453453453	1	20.0	English, Joyce A.	ProductX	Bellaire
453453453	2	20.0	English, Joyce A.	ProductY	Sugarland
333445555	2	10.0	Wong, Franklin T.	ProductY	Sugarland
333445555	3	10.0	Wong, Franklin T.	ProductZ	Houston
333445555	10	10.0	Wong, Franklin T.	Computerization	Stafford
333445555	20	10.0	Wong, Franklin T.	Reorganization	Houston
999887777	30	30.0	Zelaya, Alicia J.	Newbenefits	Stafford
999887777	10	10.0	Zelaya, Alicia J.	Computerization	Stafford
987987987	10	35.0	Jabbar, Ahmad V.	Computerization	Stafford
987987987	30	5.0	Jabbar, Ahmad V.	Newbenefits	Stafford
987654321	30	20.0	Wallace, Jennifer S.	Newbenefits	Stafford
987654321	20	15.0	Wallace, Jennifer S.	Reorganization	Houston
888665555	20	Null	Borg, James E.	Reorganization	Houston

tyas@stikom.edu

Panduan untuk Redundant Information dalam Tuples dan Update Anomalies

- Panduan 2:
  - Desain sebuah schema yang bebas dari insertion, deletion dan update anomalies.
  - Jika terjadi anomali, maka catatlah sehingga dapat dijadikan masukan untuk pengembangan selanjutnya.

tyas@stikom.edu

1.3 Null Values in Tuples

- Panduan 3:
  - Relasi seharusnya didesain untuk dapat menerima sedikit mungkin nilai NULL.
  - Atribut-atribut yang sering memiliki nilai NULL dapat dipisahkan dan disimpan dalam relasi yang berbeda (dilengkapi dengan primary key)
- Alasan untuk nilai nulls:
  - Atribut yang dimiliki tidak cocok atau tidak bisa digunakan
  - Nilai atribut tidak diketahui (mungkin ada)
  - Nilai diketahui tapi tidak tersedia

tyas@stikom.edu

1.4 Spurious Tuples

- Desain yang jelek pada relasi database akan menghasilkan hasil yang tidak benar pada saat melakukan operasi JOIN.
- The "lossless join" property digunakan untuk menjamin hasil yang dapat dipercaya pada saat melakukan melakukan operasi join.
- Panduan 4:
  - Relasi harus didesain untuk menjamin *the lossless join condition.*
  - *No spurious tuples* seharusnya diciptakan dengan melakukan *natural-join* pada setiap relasi

tyas@stikom.edu

Spurious Tuples (2)

- Terdapat dua properti yang penting untuk melakukan dekomposisi:
  a) Non-additive or losslessness of the corresponding join
  b) Preservation of the functional dependencies.

- Catatan:
  - Property (a) is extremely important and *cannot* be sacrificed.
  - Property (b) is less stringent and may be sacrificed. (terdapat penjelasan lebih lanjut).

tyas@stikom.edu

2.1 Functional Dependencies (1)

- Functional dependencies (FDs)
  - Digunakan untuk mendefiniskan ukuran formal (*formal measures) dari desain relasi yang baik*
  - Menggunakan kunci yang digunakan untuk mendefinisikan **normal forms** dari relasi
  - Merupakan **constraints** yang diturunkan dari arti (the *meaning*) dan keterkaitan (*interrelationships) dari atribut data*
- Sekumpulan atribut X secara fungsional menjelaskan (*functionally determines)* sekumpulan atribut Y jika nilai X menjelaskan nilai Y yang unik.

tyas@stikom.edu

Functional Dependencies (2)

- X -> Y holds if whenever two tuples have the same value for X, they *must have* the same value for Y
  - For any two tuples t1 and t2 in any relation instance r(R): If t1[X]=t2[X], *then* t1[Y]=t2[Y]
- X -> Y in R specifies a *constraint* on all relation instances r(R)
- Written as X -> Y; can be displayed graphically on a relation schema as in Figures. (denoted by the arrow:).
- FDs are derived from the real-world constraints on the attributes

tyas@stikom.edu

Examples of FD constraints (1)

- Social security number menjelaskan employee name
  - SSN -> ENAME
- Project number menjelaskan project name dan location
  - PNUMBER -> {PNAME, PLOCATION}
- Employee ssn dan project number menjelaskan jumlah hours per minggu yang dilakukan oleh pegawai yang bekerja pada sebuah project
  - {SSN, PNUMBER} -> HOURS

tyas@stikom.edu

Examples of FD constraints (2)

- Sebuah FD adalah property dari atribut dalam schema R
- Constraint harus mencakup setiap *relation instance r(R)*
- Jika K adalah key dari R, maka K secara functional menjelaskan semua atribut dalam R
  - (since we never have two distinct tuples with t1[K]=t2[K])

2.2 Inference Rules for FDs (1)

- Diberikan satu set dari FD F, kita dapat *mengasumsikan* (**infer)** FD tambahan yang dapat mencakup kapanpun FD di F cakup
- Armstrong's inference rules:
  - IR1. (**Reflexive**) If Y *subset-of* X, then X -> Y
  - IR2. (**Augmentation**) If X -> Y, then XZ -> YZ
    - (Notation: XZ stands for X U Z)
  - IR3. (**Transitive**) If X -> Y and Y -> Z, then X -> Z

- IR1, IR2, IR3 form a **sound** and **complete** set of inference rules
  - These are rules hold and all other rules that hold can be deduced from these

tyas@stikom.edu

Inference Rules for FDs (2)

- Beberapa asumsi tambahan yang dapat digunakan adalah:
  - **Decomposition:** If X -> YZ, then X -> Y and X -> Z
  - **Union:** If X -> Y and X -> Z, then X -> YZ
  - **Psuedotransitivity:** If X -> Y and WY -> Z, then WX -> Z (pelengkap semu)

- The last three inference rules, as well as any other inference rules, can be deduced from IR1, IR2, and IR3 (completeness property)

tyas@stikom.edu

Inference Rules for FDs (3)

- **Closure** of a set F of FDs is the set F^+ of all FDs that can be inferred from F
- **Closure** of a set of attributes X with respect to F is the set X^+ of all attributes that are functionally determined by X
- X^+ can be calculated by repeatedly applying IR1, IR2, IR3 using the FDs in F

tyas@stikom.edu

2.3 Equivalence of Sets of FDs

- Two sets of FDs F and G are **equivalent** if:
  - Every FD in F can be inferred from G, and
  - Every FD in G can be inferred from F
  - Hence, F and G are equivalent if $F^+ = G^+$
- Definition (**Covers**):
  - F **covers** G if every FD in G can be inferred from F
    - (i.e., if G^+ *subset-of* F^+)
- F and G are equivalent if F covers G and G covers F
- There is an algorithm for checking equivalence of sets of FDs

tyas@stikom.edu

2.4 Minimal Sets of FDs (1)

- Sekumpulan FD disebut **minimal** jika FD tersebut dapat memenuhi kondisi berikut ini:
  1. Setiap ketergantungan pada F memiliki single attribute pada RHS nya.
  2. Tidak dapat menghapus sembarang ketergantungan dari F dan memiliki sekumpulan ketergantungan yang equivalent dengan F.
  3. Tidak dapat mengganti sembarang ketergantungan X -> A dalam F dengan ketergantungan Y -> A, dimana Y proper-subset-of X (Y subset-of X) dan tetap memiliki sekumpulan ketergantungan yang equivalent dengan F.

tyas@stikom.edu

Minimal Sets of FDs (2)

- Setiap set dari FD yang memiliki *equivalent minimal set*
- Mungkin terdapat beberapa *equivalent minimal sets*
- Tidak ada algoritma yang sederhana untuk melakukan perhitungan pada *minimal set of FDs* yang equivalent dengan *a set F of FDs*
- Untuk menyatukan sekelompok relasi, diasumsikan bahwa dimulai dengan sekelompok ketergantungan yang memiliki minimal set.
  - E.g., see algorithms 11.2 and 11.4

LATIHAN

tyas@stikom.edu

- Diberikan relasi R dengan empat atribut ABCD. Untuk setiap kumpulan FD berikut, dengan asumsi FD hanya bergantung pada R maka identifikasi candidate key(s) untuk R.

1. $C \rightarrow D, C \rightarrow A, B \rightarrow C$
2. $B \rightarrow C, D \rightarrow A$
3. $ABC \rightarrow D, D \rightarrow A$
4. $A \rightarrow B, BC \rightarrow D, A \rightarrow C$
5. $AB \rightarrow C, AB \rightarrow D, C \rightarrow A, D \rightarrow B$