BACKGROUND

Relational data model

- a $relational\ database$ is a database whose strcuture is in the form of tables
- schemas:
  ENG(ENO,ENAME,TITLE,SAL,JNO,RESP,DUR)
  J(JNO,JNAME,BUDGET)
- tuple/record, attributes, domain, cardinality, etc.
- NO duplicate record
- records/attributes are unordered
- facilitates powerful nonprocedural languages
- primary key, secondary key, foreign key
- base relation, views (virtual relation), snapshots, query results, intermediate results, temporary relations

SAMPLE DATABASE

ENG

ENO	ENAME	TITLE	SAL	JNO	RESP	DUR
E1	J. Doe	Elect. Eng.	40000	J1	Manager	12
E2	M. Smith	Analyst	34000	J1	Analyst	24
E2	M. Smith	Analyst	34000	J2	Analyst	6
E3	A. Lee	Mech. Eng.	27000	J3	Consultant	10
E3	A. Lee	Mech. Eng.	27000	J4	Engineer	48
E4	J. Miller	Programmer	24000	J2	Programmer	18
E5	B. Casey	Syst. Ana.	34000	J2	Manager	24
E6	L. Chu	Elect. Eng.	40000	J4	Manager	48
E7	R. Davis	Mech. Eng.	27000	J3	Engineer	36
E8	J. Jones	Syst. Ana.	34000	J3	Manager	40

J

JNO	JNAME	BUDGET
J1	Instrumentation	150,000
J2	Database Development	135,000
J3	CAD/CAM	250,000
J4	Maintenance	310,000

FUNCTIONAL DEPENDENCY

- a FD is a relationship between or among attributes
- Given a relation R, attribute Y of R is $functionally$ $dependent$ on attribute X of R, denoted

  $$R.X \rightarrow R.Y$$

  iff each X-value in R has associated with it precisely one Y-value in R
- e.g., $TITLE \rightarrow SAL$
  $JNO \rightarrow JNAME$
- FD is a $semantic$ notion: recognizing FDs is part of the process of understanding what the data means
- FD is a kind of integrity constraint, which needs to be stated when a schema is declared, e.g. $entity$ $rule$ (i.e. key is non-null), $referential$ $integrity$ $rule$

NORMALIZATION

- a step-by-step reversible process of replacing a given collection of relations by successive collections in which relations have a progressively simpler and more regular structure
- aim at removing various anomalies: repetition, update, insert, delete
- 1NF, 2NF, 3NF, BCNF, etc.

NORMALIZED RELATIONS

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng.
E2	M. Smith	Analyst
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Ana.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Ana.

ENO	JNO	RESP	DUR
E1	J1	Manager	12
E2	J1	Analyst	24
E2	J2	Analyst	6
E3	J3	Consultant	10
E3	J4	Engineer	48
E4	J2	Programmer	18
E5	J2	Manager	24
E6	J4	Manager	48
E7	J3	Engineer	36
E8	J3	Manager	40

JNO	JNAME	BUDGET
J1	Instrumentation	150,000
J2	Database Development	135,000
J3	CAD/CAM	250,000
J4	Maintenance	310,000

TITLE	SAL
Elect. Eng.	40000
Syst. Anal.	34000
Mech. Eng.	27000
Programmer	24000

RELATIONAL ALGEBRA

a set of operators on relations: selection, projection, union, set difference and cartesian product

- selection/restriction: $\sigma_{TITLE=\text{`}Elect.Eng.'}(E)$

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng.
E6	L. Chu	Elect. Eng.

- projection: $\pi_{JNO,BUDGET}(J)$

JNO	BUDGET
J1	150,000
J2	135,000
J3	250,000
J4	310,000

RELATIONAL ALGEBRA (CONT)

- equi-join: $E \bowtie_{E.TITLE=S.TITLE} S$

ENO	ENAME	TITLE	SAL
E1	J. Doe	Elect. Eng.	40000
E2	M. Smith	Analyst	34000
E3	A. Lee	Mech. Eng.	27000
E4	J. Miller	Programmer	24000
E5	B. Casey	Syst. Ana.	34000
E6	L. Chu	Elect. Eng.	40000
E7	R. Davis	Mech. Eng.	27000
E8	J. Jones	Syst. Ana.	34000

- semijoin: $E \ltimes_{E.TITLE=S.TITLE} S$

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng.
E2	M. Smith	Analyst
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Ana.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Ana.

– Note: $R1 \ltimes R2 = \pi_{R1}(R1 \bowtie R2)$

RELATIONAL CALCULUS

- a $nonprocedural$ language based on the theoretical foundation of first-order predicate logic
- tuple relational calculus:

$$\{t|F(t)\}$$

  where F(t) is a well-formed formula
- domain relational calculus:

$$\{(x_1, x_2, \ldots, x_k)|F(x_1, x_2, \ldots, x_k)\}$$

STRUCTURED QUERY LANGUAGE (SQL)

- a standard
- most commerical systems support the language
- based on relational calculus and algebra
- Examples:

```
SELECT   *
FROM     E
WHERE    E.TITLE = 'Elect. Eng.'

SELECT   JNO, BUDGET
FROM     J

SELECT   ENO, ENAME, TITLE, SAL
FROM     E, S
WHERE    E.TITLE = S.TITLE
```

RELATIONAL DBMS