Time-Dependent Data

(Chapter 10)

1

Temporal Data Categories

- **Type 1:** No history (new attribute values written over old ones)
- **Type 2:** Complete history (every previous attribute value must be retained)
- **Type 3:** Limited history (only a finite number of previous values is retained)
- **Hybrids**

2

Which Model Should Include Temporal Data?

- If requirements are profound and/or obvious (e.g. airline scheduling, OLAP databases), include in Conceptual Model
- If generally invisible to business users (e.g. audit columns, log tables), include in Physical Model
- In most other cases, add to Logical Model

3

Time-Dependent Data Requirements

- Must be able to handle:
  - Events of the past (but seldom full history for everything in the database)
  - Status at any given date in the past
  - Future events
  - Different time zones (perhaps)
- Audit trails are often required to show what was changed, by whom, when and why

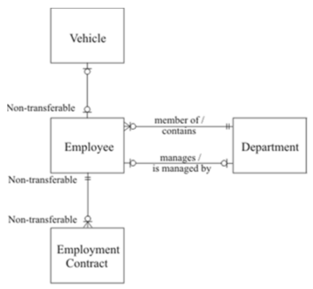
4

Adding History to Data Structures

- Adding History:
  - Changes 1:1 relationships to 1:M
  - Changes 1:M relationships to M:N
  - **EXCEPT** non-transferable relationships

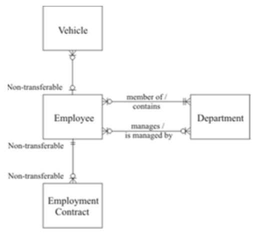
5

Conceptual Model Without History



6

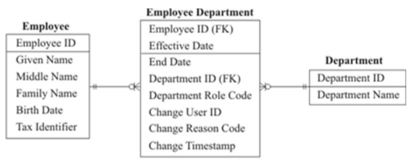
Conceptual Model With History



- Employee-Vehicle now 1:M
- Department-Employee (member) now M:N
- Department_Employee (manages) now M:N
- Non-transferable "ends" of relationships not changed

7

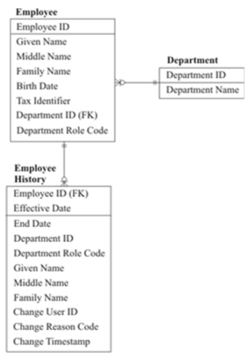
Logical Model with Effective Date in the Primary Key



- Timestamp can be used instead of Date if required
- End Date not part of the PK (does not add to uniqueness)
- A surrogate key is always an option, but make sure you define a unique constraint on the "natural" key
- Set a standard for default begin and end dates values

8

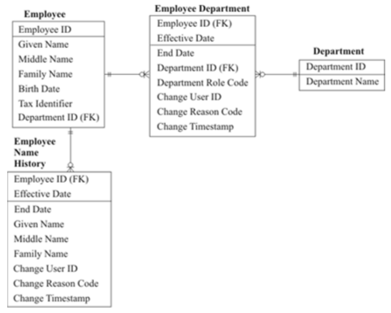
The History Table Alternative



- Only the current row appears in Employee
- It may be desirable to also include the current row in Employee History
- Values that won't change except for corrections (e.g. Birth Date) can be left off the history table
- Effective-dated and history table approach can be combined into a hybrid (next slide)

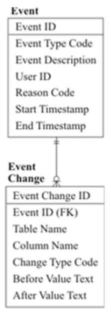
9

Hybrid Including History Tables and Effective-dated Tables



10

The Change Log Alternative



- Event records events that change data (e.g. an employee name change)
- Event Change has a record for each data value that changed during the event (e.g. First Name, Middle Name, Last Name)
- Consider generalized audit logging software (embedded in DBMS or independent of the DBMS)
- Consider subtyping events that have diverse requirements

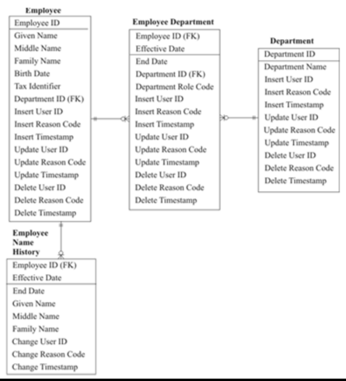
11

Handling Deletions

- Deleted records cannot be purged if there are requirements for history
- Alternative solutions (use 1 or more):
  - Set the End Date to show logical deletion
  - Include a Status Code attribute with a data value for “deleted”
  - Mark current row with a code or flag
  - Copy record to audit log before purging
  - Audit columns specific to deletion

12

Audit Columns for Insert, Update, and Delete



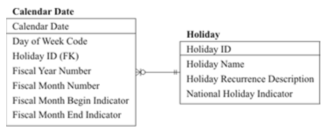
13

Archiving / Purging

- When to archive is a business unit decision
- When to purge is a legal department issue

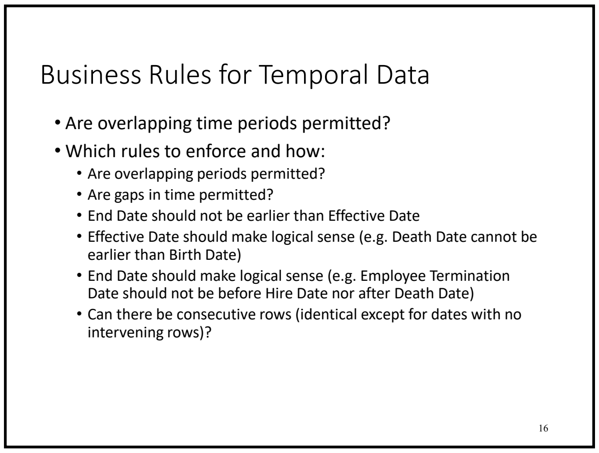
14

Calendar Data Structures



- Calendar Date has a row for every date (day) of interest to the business
- Avoid the temptation to make every date in every table a foreign key to the Calendar Date entity

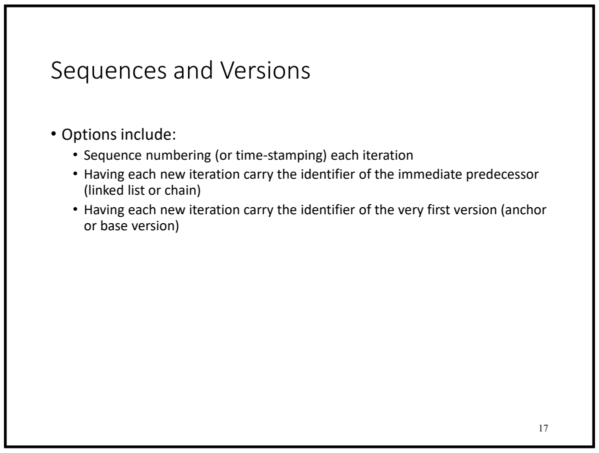
15



Business Rules for Temporal Data

- Are overlapping time periods permitted?
- Which rules to enforce and how:
  - Are overlapping periods permitted?
  - Are gaps in time permitted?
  - End Date should not be earlier than Effective Date
  - Effective Date should make logical sense (e.g. Death Date cannot be earlier than Birth Date)
  - End Date should make logical sense (e.g. Employee Termination Date should not be before Hire Date nor after Death Date)
  - Can there be consecutive rows (identical except for dates with no intervening rows)?

16



Sequences and Versions

- Options include:
  - Sequence numbering (or time-stamping) each iteration
  - Having each new iteration carry the identifier of the immediate predecessor (linked list or chain)
  - Having each new iteration carry the identifier of the very first version (anchor or base version)

17