

An Analysis of Table Constraints in SQL2  
based on the  
Entity-Relationship Model

**Marco A. Casanova**  
**Alexandre P. de Carvalho**  
**Lorenzo F.G.G.M. Ridolfi**

Rio Scientific Center  
IBM Brazil  
Rio de Janeiro, Brazil

**Alberto H.F. Laender**

Computer Science Department  
Federal University of Minas Gerais  
Belo Horizonte, Brazil

# Topics

---

- Preliminaries
- Summary of SQL2 Table Constraints
- The Two-Level ER Model
- Mapping of ER Schemas into SQL2
  - Mapping of Level 0 ER Schemas
  - Mapping of Level 1 ER Schemas
- Optimization
- Conclusions

# Preliminaries

---

- Goal of the Work:
  - Analysis of SQL2 table constraints in the light of the ER model
- Method Used:
  - SQL2 proposal distinguishes two subsets of the language:
    - Entry SQL
    - Intermediate SQL
  - Analysis is based on a two-level ER model
- Contributions:
  - Careful analysis of the mapping of ER schemas into SQL2
  - Suggestion of a minor extension to SQL2 that increases the ability to obtain optimized representations of ER schemas

## Summary of SQL2 Table Constraints

---

- Table constraint definitions:
  - unique constraint definitions
  - referential constraint definitions
  - check constraint definitions
- Constraint checking modes:
  - immediate
  - deferred

For Entry SQL, all constraints are always immediately checked

# Summary of SQL2 Table Constraints

---

- Unique constraint definitions:

- PRIMARY KEY (not null implicit)
- UNIQUE (may admit null values)

Entry SQL requires that all attributes of a primary or alternate key do not allow null values

- Referential constraint definitions:

- match type: (omitted)/FULL
- delete and update rules:  
CASCADE/SET NULL/SET DEFAULT

Entry SQL does not support either delete or update rules and Intermediate SQL does not support update rules

- Check constraint definitions:

- specify conditions that the rows of a table must satisfy

Entry and Intermediate SQL allow only simple restrictions

# The Two-Level ER Model

---

- Level 0 ER model:
  - entity schemes
  - relationship schemes
  - hierarchies of specialization for entity schemes
- Level 1 ER model:
  - totality on relationship schemes
  - multiple specialization of entity schemes
  - propagation of deletions and updates

## Mapping of Level 0 ER Schemas

---

- Basic approach:
  - One-to-one mapping which captures references between ER objects through referential constraints
- Example:



```
create table T_EMP  
(ID char(5) not null,  
NAME char (25),  
primary key (ID))
```

```
create table T_DEPT  
(D# char(3) not null,  
LOCATION char(20),  
primary key (D#))
```

```
create table T_WORK  
(ID char(5) not null,  
D# char(3),  
primary key (ID),  
foreign key (ID) references T_EMP,  
foreign key (D#) references T_DEPT)
```

# Mapping of Level 0 ER Schemas

## Conclusions

---

- Mapping of level 0 ER schemas DOES NOT require:
  - deferred checking
  - unique constraint definitions with support for:
    - alternate keys that allow null values
  - referential constraint definitions with support for:
    - references to alternate keys
    - delete rules
    - SET NULL and SET DEFAULT update rules
    - match type
  - check constraint definitions
- If we do not allow updates on keys, Entry SQL suffices to handle the mapping of level 0 ER schemas

# Mapping of Level 1 ER Schemas

## Handling of Totality

---

- Case 1: R is total and functional (N:1) on E
  - Completely handled by a referential constraint definition
- Case 2: R is total on E but not functional
  - Requires a check constraint definition

# Mapping of Level 1 ER Schemas

## Handling of Multiple Specialization

---

- Requires the use of references to alternate keys

A(K)      B(L)

C(K)      D(K,L)      E(L)

F(K)      G(L)

```
create table T_D
(K char(20) not null,
L char(20) not null,
primary key (K),
unique (L),
foreign key (K) references T_A
on update cascade,
foreign key (L) references T_B
on update cascade)
```

```
create table T_G
(L char(20) not null,
primary key (L),
foreign key (L) references T_D(L)
on update cascade)
```

# Mapping of Level 1 ER Schemas

## Conclusions

---

- Mapping of level 1 ER schemas DOES NOT require:
  - unique constraint definitions with support for:
    - alternate keys that allow null values
  - referential constraint definitions with support for:
    - SET NULL and SET DEFAULT update and delete rules
    - match type
- but it requires:
  - deferred checking
  - referential constraint definitions with support for:
    - references to alternate keys
    - the CASCADE delete rule
    - the CASCADE update rule, if we permit updates on keys or identifiers
  - check constraint definitions
- If we do not permit updates on keys or identifiers, Intermediate SQL suffices to handle the mapping of level 1 ER schemas; but Entry SQL does not suffice, since the mapping of level 1 ER schemas requires deferred checking and delete rules

# Optimization of SQL2 Schemas

---

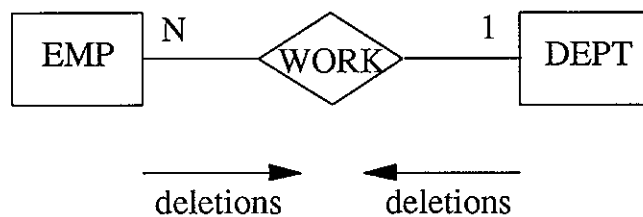
- Motivation:
  - one-to-one relational representations of ER schemas are straightforward to obtain, but they contain a potentially large number of inter-relation references that are expensive to check for violations
- Basic optimization heuristics:
  - to collapse a relationship scheme  $R$  into an entity scheme  $E$ , when  $R$  is functional on  $E$
  - to collapse an entity scheme  $F$  into an entity scheme  $E$ , when  $F$  specializes  $E$

# Optimization

## Example 1: SET NULL delete rule

---

- ER schema:



- SQL2 schema:

```
create table T_EMP*  
  (ID char(5) not null,  
   NAME char (25),  
   D# char(3),  
   primary key (ID),  
   foreign key (D#) references T_DEPT  
   on delete set null)
```

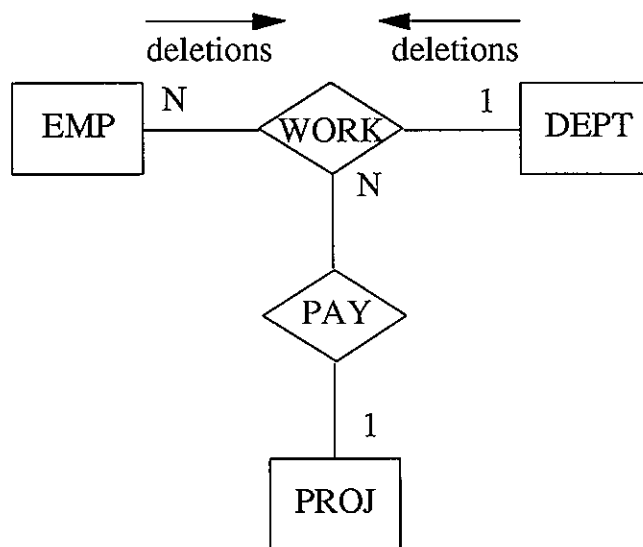
```
create table T_DEPT  
  (D# char(3) not null,  
   LOCATION char(20),  
   primary key (D#))
```

# Optimization

## Example 2: check constraint definition

---

- ER schema:



- SQL2 schema:

```
create table T_EMP*
  (same as before)
```

```
create table T_DEPT
  (same as before)
```

```
create table T_PROJ
  (P# char(3) not null,
  DURATION char(6),
  primary key (P#))
```

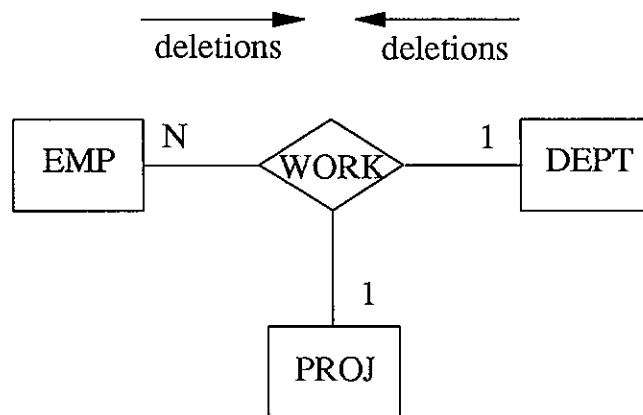
```
create table T_PAY
  (ID char(5) not null,
  P# char(3) not null,
  primary key (ID),
  foreign key (P#) references T_PROJ
  check ID match (select ID
    from T_EMP*
    where D# is not null))
```

# Optimization

## Example 3: SET NULL propagation

---

- ER schema:



- Tentative SQL2 schema:

```
create table T_EMP*
  (ID char(5) not null,
  NAME char (25),
  D# char(3),
  P# char(3),
  primary key (ID),
  foreign key (D#) references T_DEPT
  on delete set null,
  foreign key (P#) references T_PROJ)
```

```
create table T_DEPT
  (D# char(3) not null,
  LOCATION char(20),
  primary key (D#))
```

```
create table T_PROJ
  (P# char(3) not null,
  DURATION char(20),
  primary key (P#))
```

# Optimization

## Example 3: SET NULL propagation

---

- Problem:
  - the SET NULL option propagates nulls only to attributes of the foreign key

T_EMP*				T_DEPT		T_PROJ	
ID	NAME	D#	P#	D#	LOCATION	P#	DURATION
e1	John	d1	p1	d1	b1	p1	t1
e2	Mary	d2	p2	d2	b2	p2	t2

- Solution proposed:
  - the SET NULL option should be modified to allow the propagation of nulls to attributes outside the foreign key

foreign key (D#) references T\_DEPT  
on delete set null (D#,P#)

# Optimization

## Conclusions

---

- Optimized mapping requires:
  - deferred checking
  - unique constraint definitions with support for:
    - alternate keys that allow null values
  - referential constraint definitions with support for:
    - references to alternate keys
    - the CASCADE and SET NULL delete rules
    - the CASCADE update rule, if we permit updates on keys or identifiers
  - check constraint definitions
- The SET NULL option should be modified to cover a more generalized form of propagation of nulls

## Conclusions

---

- Result of the analysis:
  - Classification of the table constraint features of SQL2

Level 0 ER schemas	Entry SQL
Level 1 ER schemas	Intermediate SQL (deferred checking + deletion rules)
Level 1 ER schemas (optimized mapping)	Full SQL (alternate keys with nulls)

- SET DEFAULT and match type are of limited utility to the mapping of ER schemas to SQL2
  - SET NULL option should be modified to allow the propagation of nulls to attributes outside the foreign key
- The analysis can be extended to cover other ER concepts, such as multivalued attributes and more complex forms of specialization/generalization