CS120 ProblemSet1

Yuqing Wang

September 2025

1 Schema Refinement

(a)

No, because the primary key is {`EMP_ID, REC_ID`}. For example, in a functional dependency like `EMP_ID -> ROLE`, EMP_ID is not a super key of this table.

(b)

- Functional dependencies:
  - `EMP_ID -> EMP_NAME, ROLE`
  - `REC_ID -> REC_NAME, REC_ARTIST, PRICE`
- Step 1 (using `EMP_ID -> EMP_NAME, ROLE`):
  - `R1(EMP_ID, EMP_NAME, ROLE)`
  - `R2(EMP_ID, REC_ID, REC_NAME, REC_ARTIST, PRICE)`
- Step 2 (in `R2`, using `REC_ID -> REC_NAME, REC_ARTIST, PRICE`):
  - `R3(REC_ID, REC_NAME, REC_ARTIST, PRICE)`
  - `R4(EMP_ID, REC_ID)`
- **Final BCNF relations:**
  - `Employee(EMP_ID, EMP_NAME, ROLE)`
  - `Record(REC_ID, REC_NAME, REC_ARTIST, PRICE)`
  - `Sales(EMP_ID, REC_ID)`

emp_id	emp_name	role
1	Bob	Cashier
2	Lupe	Manager
3	Sara	Asst. Manager
4	Geoff	Security
5	Dakota	Cashier

Figure 1: Employee

rec_id	rec_name	rec_artist	price
1	Rocky Mountain High	John Denver	10
5	Back to the Country	Loretta Lynn	15
3	Jolene	Dolly Parton	12
2	Rocky Mountain High	John Denver	10
4	22	Taylor Swift	10

Figure 2: Record

emp_id	rec_id
1	1
2	5
3	3
4	2
5	4

Figure 3: Sales

2 Schema Design

(a)

Venue

- Attributes: `venue_id`, `name`, `address`
- Primary Key: `venue_id`
- Constraints: `name` and `address` must be non-null
- Justification: Each venue has a unique identifier and stores basic information about the location.

Section

- Attributes: `section_id`, `venue_id`, `name`, `capacity`
- Primary Key: `section_id`
- Constraints: `capacity` must be strictly positive; `venue_id` is a foreign key referencing Venue
- Justification: Each section belongs to exactly one venue, and its seating capacity must be valid.

Performance

- Attributes: `performance_id`, `artist`, `show_date`, `venue_id`
- Primary Key: `performance_id`
- Constraints: `show_date` must be non-null; `venue_id` is a foreign key referencing Venue
- Justification: Each performance is uniquely identified and must take place at exactly one venue.

TicketPrice

- Attributes: `performance_id`, `section_id`, `price`
- Primary Key: Composite key (`performance_id`, `section_id`)
- Constraints: `price` must be non-negative; `performance_id` references Performance; `section_id` references Section
- Justification: The price of a ticket depends on both the performance and the section within the venue.

(b)

- If the ticket price for a section is not fixed but rather dynamic (for example, early-bird tickets versus regular tickets), this schema would not be able to support it.
- If a performance involves multiple artists performing together, and each artist has a separate ticket price, this schema would also not be compatible.

3 Relational Algebra and SQL

(a) Find the wids of workers who made ≥1 type of toys in category 'LEGO'

Relational Algebra:

$\pi_{wid}\big(\sigma_{category=\text{'LEGO'}}(Catalog)\big)$

SQL:

```
SELECT DISTINCT wid
FROM Catalog
WHERE category = 'LEGO';
```

(b) Find the wids of workers who produced only 'LEGO'

Relational Algebra:

$\pi_{wid}(Catalog) - \pi_{wid}(\sigma_{category\neq\text{'LEGO'}}(Catalog))$

SQL:

```
SELECT wid
FROM Catalog
GROUP BY wid
HAVING COUNT(DISTINCT category) = 1
   AND MIN(category) = 'LEGO';
```

(c) Find the names of workers and the customers their toys are sold to

Relational Algebra:

$\pi_{wname,cname}(Worker \bowtie Catalog \bowtie SalesRecord \bowtie Customer)$

SQL:

```
SELECT DISTINCT w.wname, c.cname
FROM Worker w
JOIN Catalog ca ON w.wid = ca.wid
JOIN SalesRecord s ON ca.iid = s.iid
JOIN Customer c ON s.cid = c.cid;
```

(d) Find each customer that bought at least 10 different toys

Relational Algebra:

Not expressible in relational algebra without aggregation

SQL:

```
SELECT c.cname
FROM Customer c
JOIN SalesRecord s ON c.cid = s.cid
GROUP BY c.cid, c.cname
HAVING COUNT(DISTINCT s.iid) >= 10;
```

(e) Find the name(s) of the most expensive toy(s) bought by Mrs Claus

Relational Algebra:

Not expressible in relational algebra without the MAX operator.

SQL:

```
-- All toys she bought before
WITH ClausToys AS (
    SELECT ca.iname, s.unit_price
    FROM SalesRecord s
    JOIN Catalog ca ON s.iid = ca.iid
    JOIN Customer c ON s.cid = c.cid
    WHERE c.cname = 'Mrs Claus'
)

-- Find the most expensive ones
SELECT iname
FROM ClausToys
WHERE unit_price = (SELECT MAX(unit_price) FROM ClausToys);
```