NAME: QUINUEL TABOT NDIP-AGBOR

MATRICULE: FE21A300

SPECIALISATION: SOFTWARE ENGINEERING

COURSE: ADVANCED DATABASES

ASSIGNMENT 1

2.1. Appropriate Primary Keys

Explanation:

- A primary key uniquely identifies each tuple (row) in a relation (table).
- It should be minimal (no redundant attributes) and ensure data integrity.

Solutions:

- `employee`: Both `person_name` and `(assumed) ID` could be primary keys. Each uniquely identifies an employee. We'd typically choose a numeric ID for efficiency.
- `works`: The primary key should be the combination of `(ID, company_name)`. An employee can work for multiple companies, and `ID` alone wouldn't suffice.
- `company`: The primary key is likely `company_name`. Assuming companies are unique entities, this attribute uniquely identifies each company.

2.2. Foreign Key Constraint Violation Examples

Explanation:

- A foreign key references a primary key in another relation. It ensures data consistency.

Violation: Inserting a department name in `works.company_name` that doesn't exist in `company.company_name`.

Invalid Insert: `INSERT INTO works VALUES ('John Doe', 'InvalidCompany', 80000);`

Explanation: Since 'InvalidCompany' isn't present in the `company` table, there's no corresponding department for John Doe to work in. This violates the foreign key constraint between `works.company_name` and `company.company_name`.

Violation: Deleting a company from `company` that has employees working for it in `works`.

Invalid Delete: `DELETE FROM company WHERE company_name = 'BigBank';`

Explanation: This deletion would leave orphaned entries in `works` where employees previously worked for 'BigBank'. The foreign key constraint is violated because these entries would reference a non-existent company.

2.3. Time Slot Primary Key

Explanation:

- A primary key should uniquely identify a tuple within a relation.

Solution:

- `day` and `start_time` are part of the primary key because they uniquely identify a specific time slot within a week (e.g., Tuesday 10:00 AM).
- `end_time` isn't included because a time slot can have multiple occurrences (e.g., Tuesday 10:00 AM could be used for multiple classes). Adding it wouldn't guarantee uniqueness of the time slot.

2.4. Name as Superkey (or Primary Key)

Explanation:

- A superkey is a set of attributes that uniquely identifies all tuples in a relation.
- A primary key is a minimal superkey (no redundant attributes for uniqueness).

Answer:

No, you cannot conclusively determine that `name` can be a superkey (or primary key) based on the limited information in Figure 2.1. While the current data doesn't show duplicate names, it's possible to have future insertions with the same name. Using `name` alone could lead to ambiguity in identifying instructors.

2.5. Cartesian Product and Selection

Explanation:

- Relational algebra operations manipulate relations to retrieve specific data.

Query: `σsid=ID(student × advisor)`

Breakdown:

1. **Cartesian product (student × advisor):** This creates a temporary relation containing all possible combinations of tuples from the `student` and `advisor` relations. Imagine joining each row from `student` with every row from `advisor`.
2. **Selection (σsid=ID):** This operation filters the temporary relation, keeping only tuples where the `sid` attribute from the `student` table matches the `ID` attribute from the `advisor` table. Essentially, it finds students who are advised by specific advisors. This is a selection based on a join condition.

2.6. Relational Algebra Expressions

a. Employees in Miami:

```
π person_name (σ city='Miami' (employee))
```

Explanation:

1. **σ city='Miami' (employee):** This selection (`σ`) filters the `employee` relation to include only employees where the `city` attribute is equal to 'Miami'. It keeps rows with matching city values.
2. **π person_name (...):** The `π` (projection) operation then selects only the `person_name` attribute from the result of the previous selection. It projects a specific column from the filtered data.

b. Employees with Salary > $100000:

```
π person_name (σ salary > 100000 (works))
```

Explanation:

1. **σ salary > 100000 (works):** This selection (`σ