Fiora Xu

Given the entities:

1. **Driver** (driver_id, name, contactInfo, basepay_id)
2. **Stops** (stop_id, route_id, stop_type)
3. **TimeTaken** (timeTaken_id, route_id, driver_id, date, totalTime)
4. **Route** (route_id, difficulty)
5. **AmountOfPackages** (stop_id, date, total_packages)
6. **BasePay** (basepay_id, base_rate, package_threshold, extra_rate_per_package)
7. **Performance**(performance_id, driver_id, date, stops_completed, packages_delivered, time_efficiency)
8. **Bonuses**(bonus_id, driver_id, date, reason, amount)

Functional Dependencies (FDs)

FDs for each entity:

1. Driver: driver_id → name, contactInfo, basepay_id
2. Stops: stop_id → route_id, stop_type
3. TimeTaken: timeTaken_id → route_id, driver_id, date, totalTime
4. Route: route_id → difficulty
5. AmountOfPackages: packages_id → stop_id, date, total_packages
6. BasePay: basepay_id → base_rate, package_threshold, extra_rate_per_package
7. Performance: performance_id → driver_id, date, route_id, stops_completed, packages_delivered, time_efficiency
8. Bonuses: bonus_id → driver_id, date, reason, amount

Relational Schema in BCNF

1. Driver(driver_id PK, name, contactInfo, basepay_id FK)
   - FD: driver_id → name, contactInfo, basepay_id
   - All attributes are functionally dependent on the primary key, so it's in BCNF.

2. Stops(stop_id PK, route_id FK, stop_type)
   - FD: stop_id → route_id, stop_type
   - In BCNF as stop_id is a superkey.

3. TimeTaken(timeTaken_id PK, route_id FK, driver_id FK, date, totalTime)
   - FD: timeTaken_id → route_id, driver_id, date, totalTime

- timeTaken_id is a superkey, satisfying BCNF.

4. Route(route_id PK, difficulty)
- FD: route_id → difficulty
- route_id is a superkey, so it's in BCNF.

5. AmountOfPackages(packages_id PK, stop_id FK, date, total_packages)
- FD: packages_id → stop_id, date, total_packages
- packages_id is a superkey, in BCNF.

6. BasePay(basepay_id PK, base_rate, package_threshold, extra_rate_per_package)
- FD: basepay_id → base_rate, package_threshold, extra_rate_per_package
- basepay_id is a superkey, so it's in BCNF.

7. Performance(performance_id PK, driver_id FK, date, route_id FK, stops_completed, packages_delivered, time_efficiency)
- FD: performance_id → driver_id, date, route_id, stops_completed, packages_delivered, time_efficiency
- performance_id is a superkey, in BCNF.

8. Bonuses(bonus_id PK, driver_id FK, date, reason, amount)
- FD: bonus_id → driver_id, date, reason, amount
- bonus_id is a superkey, in BCNF.

Relationships

Based on the relational schema and the functional dependencies provided, here are the relationships between the entities:

1. Driver to BasePay: One-to-One (1:1)
- Each `Driver` is assigned exactly one `BasePay` through `basepay_id`. Since `basepay_id` is unique for each `Driver`, this forms a one-to-one relationship.

2. Driver to Performance: One-to-Many (1:N)
- A `Driver` can have multiple `Performance` records over time, indicated by `performance_id`. Each `Performance` record is unique to a `Driver` on a specific date, route, etc.

3. Driver to Bonuses: One-to-Many (1:N)

- A `Driver` can receive multiple `Bonuses`, as each `bonus_id` is unique and can be awarded to the `Driver` for various reasons on different dates.

4. Driver to TimeTaken: One-to-Many (1:N)
- A `Driver` can have multiple `TimeTaken` records, each representing the time taken for different routes on different dates.

5. Route to Stops: One-to-Many (1:N)
- A `Route` can have multiple `Stops`, as stops are parts of a route. Each `Stop` is uniquely identified and associated with a `Route`.

6. Route to TimeTaken: One-to-Many (1:N)
- A single `Route` can be associated with multiple `TimeTaken` records, as different drivers might take the same route on different dates.

7. Stop to AmountOfPackages: One-to-Many (1:N)
- A `Stop` can have multiple `AmountOfPackages` records, as the number of packages delivered can vary by date.

8. Route to Performance: One-to-Many (1:N)
- A `Route` can be associated with multiple `Performance` records through different drivers and dates.