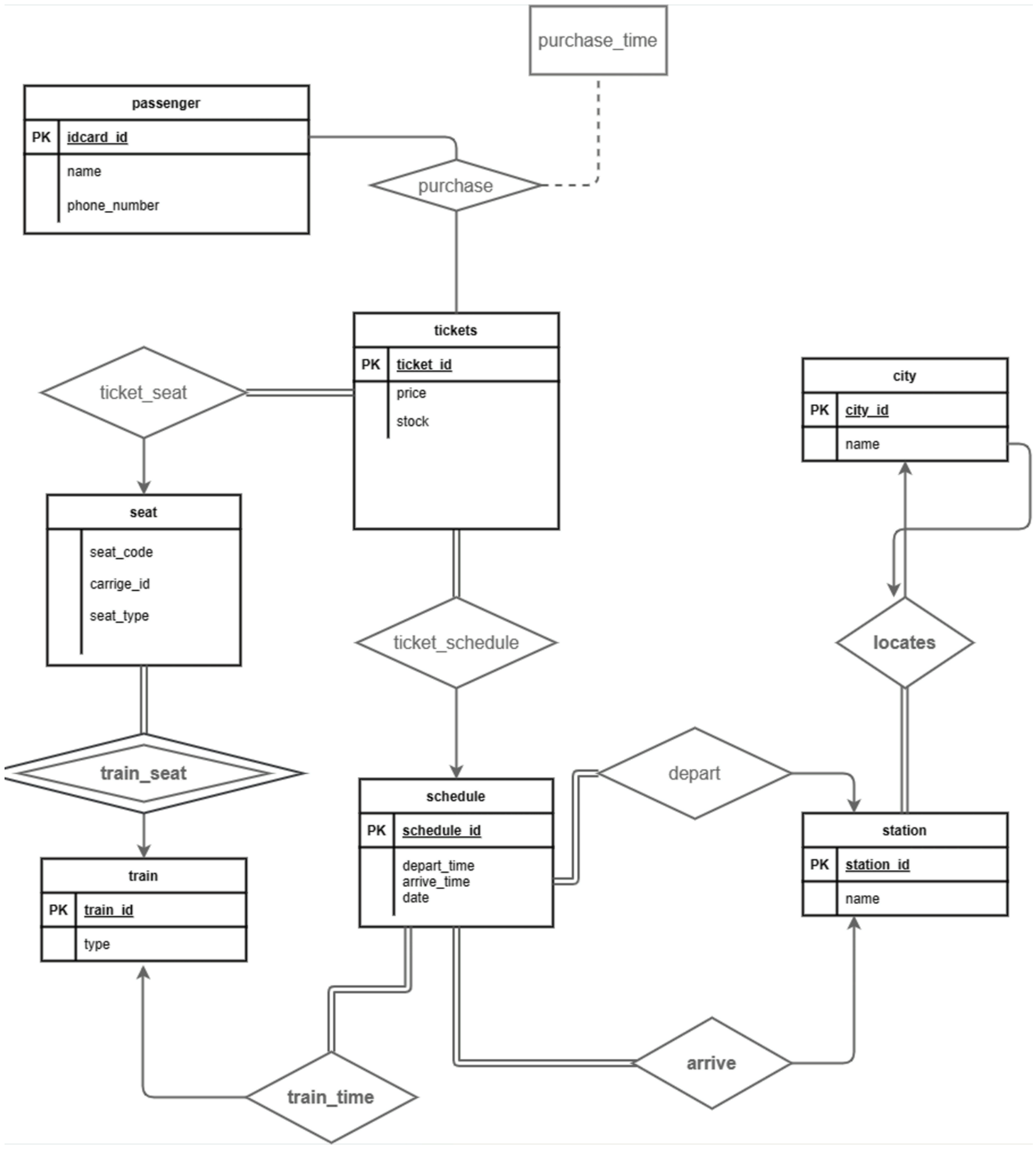
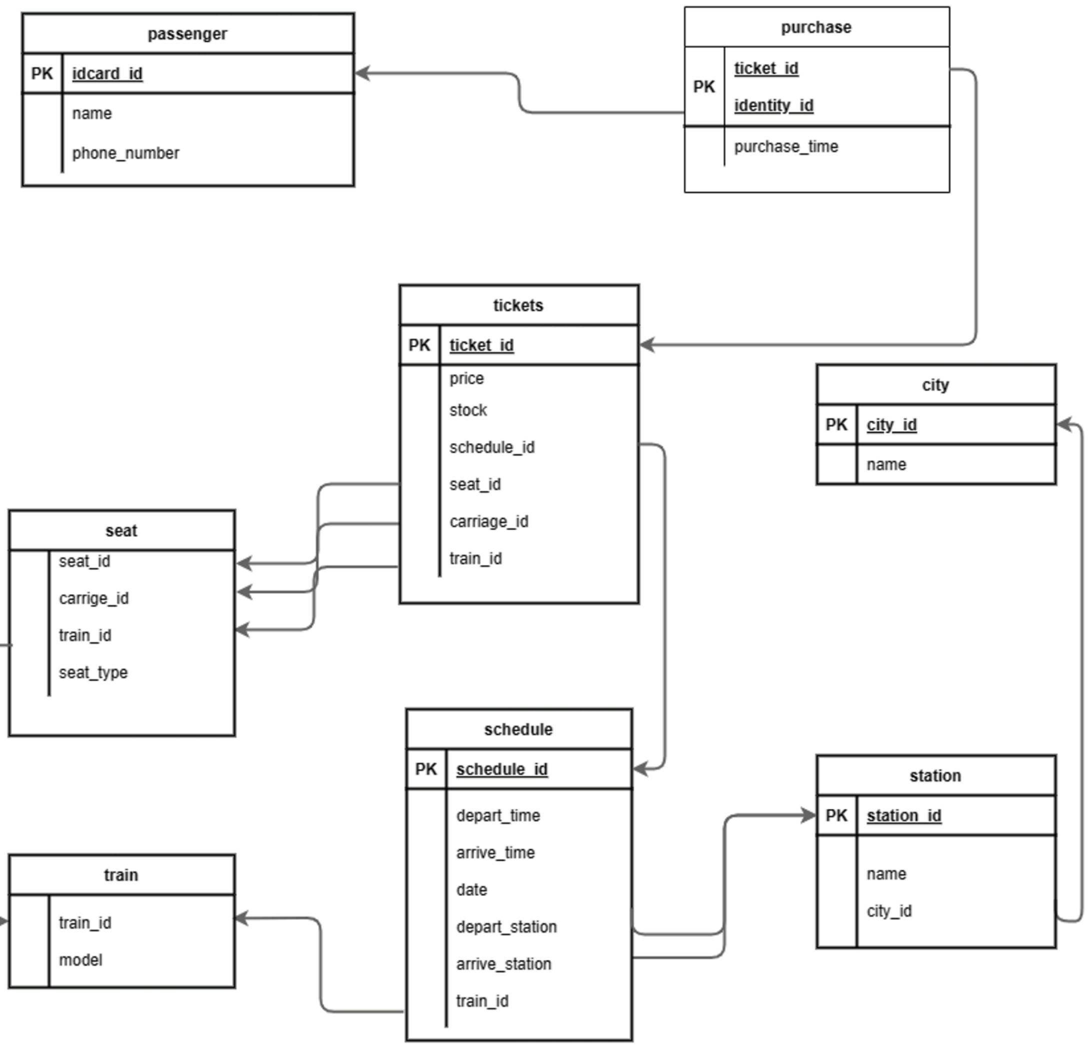
- E-R 图


purchase_time
passenger
PK idcard_id
name
phone_number
purchase
tickets
PK ticket_id
price
stock
ticket_seat
seat
seat_code
carrige_id
seat_type
train_seat
train
PK train_id
type
ticket_schedule
schedule
PK schedule_id
depart_time
arrive_time
date
depart
city
PK city_id
name
locates
station
PK station_id
name
train_time
arrive

- 关系模式：



- 建立表格

```
CREATE TABLE city (
    city_id INT PRIMARY KEY,
    name VARCHAR(20) NOT NULL COMMENT '城市名称'
);

CREATE TABLE station (
    station_id INT PRIMARY KEY,
    city_id INT,
    name VARCHAR(20) NOT NULL COMMENT '车站名称',
    FOREIGN KEY (city_id) REFERENCES city(city_id),
    INDEX (city_id)
);

CREATE TABLE train (
    train_id INT PRIMARY KEY,
    model VARCHAR(20) NOT NULL COMMENT '火车型号'
);

CREATE TABLE schedule (
    schedule_id INT PRIMARY KEY,
    depart_time TIME,
    arrive_time TIME,
    date DATE,
    depart_station INT,
    terminate_station INT,
    train_id INT,
    FOREIGN KEY (depart_station) REFERENCES station(station_id),
    FOREIGN KEY (terminate_station) REFERENCES station(station_id),
    FOREIGN KEY (train_id) REFERENCES train(train_id),
    INDEX (train_id),
    INDEX (depart_station),
    INDEX (terminate_station)
);

CREATE TABLE seat (
    seat_id INT AUTO_INCREMENT PRIMARY KEY,
    carriage_id INT,
    train_id INT,
    seat_class ENUM('A', 'B', 'C') NOT NULL COMMENT '座位等级',
    UNIQUE KEY (seat_id, carriage_id, train_id),
    FOREIGN KEY (train_id) REFERENCES train(train_id),
    INDEX (train_id),
    INDEX (carriage_id)
);

CREATE TABLE ticket (
    ticket_id INT AUTO_INCREMENT PRIMARY KEY,
    price DECIMAL(5, 2),
    available BOOLEAN,
    schedule_id INT,
    carriage_id INT,
    seat_id INT,
    train_id INT,
    FOREIGN KEY (schedule_id) REFERENCES schedule(schedule_id),
    FOREIGN KEY (seat_id, carriage_id, train_id) REFERENCES seat(seat_id,
carriage_id, train_id),
    INDEX (schedule_id),
    INDEX (carriage_id),
    INDEX (train_id)
);

CREATE TABLE passenger (
    identity_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(20) NOT NULL COMMENT '乘客姓名'
);

CREATE TABLE purchases (
    purchase_id INT AUTO_INCREMENT PRIMARY KEY,
    ticket_id INT,
    identity_id INT,
    purchase_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (ticket_id) REFERENCES ticket(ticket_id),
    FOREIGN KEY (identity_id) REFERENCES passenger(identity_id),
    INDEX (ticket_id),
    INDEX (identity_id)
);
```

- 具体应用场景
    - 查询某日从城市A到城市B的所有动车信息（包括各类座位的剩票情况），按发车时间排列

```sql
WITH schedule_info AS (
    SELECT s.schedule_id, s.train_id, s.depart_time, s.arrive_time
    FROM schedule s
    INNER JOIN station a ON s.depart_station = a.station_id
    INNER JOIN station b ON s.terminate_station = b.station_id
    WHERE s.date = some_date
        AND a.city_id = "A"
        AND b.city_id = "B"
),
ticket_info AS (
    SELECT t.schedule_id, s.seat_class, COUNT(*) AS available_num
    FROM ticket t
    INNER JOIN seat s ON t.seat_id = s.seat_id AND t.carriage_id =
s.carriage_id AND t.train_id = s.train_id
    WHERE t.schedule_id IN (SELECT schedule_id FROM schedule_info)
        AND t.available = TRUE
    GROUP BY t.schedule_id, s.seat_class
)
SELECT *
FROM schedule_info
JOIN ticket_info USING (schedule_id)
ORDER BY depart_time;
```

- 订票时，系统要知道每个座位的卖出或空余情况，以便订票

```sql
SELECT seat_id, carriage_id, available
FROM seat
WHERE train_id = some_train_id;
```

- 查询某日某车站的所有车次

```sql
SELECT schedule_id
FROM schedule
WHERE date = some_date
    AND (depart_station = some_station_id OR terminate_station =
some_station_id);
```

- 查询某日某次列车的行程表（从起点到终点所有车站的到达，发车时间，停留时间）

```sql
WITH depart_info AS (
    SELECT depart_station AS station, depart_time
    FROM schedule
    WHERE date = some_date
    AND train_id = some_train_id
),
arrive_info AS (
    SELECT terminate_station AS station, arrive_time
    FROM schedule
    WHERE date = some_date
    AND train_id = some_train_id
)
SELECT station, arrive_time, depart_time, TIMEDIFF(depart_time,
arrive_time) AS stay_time
FROM depart_info
```

```
JOIN arrive_info USING (station);
```

- 查询某躺列车和某位乘客同乘一个车厢的所有乘客

```
WITH passenger_info AS (
    SELECT p.identity_id, t.train_id, t.carriage_id
    FROM passenger p
    JOIN purchases pu ON p.identity_id = pu.identity_id
    JOIN ticket t ON pu.ticket_id = t.ticket_id
    WHERE p.identity_id = some_identity_id AND t.train_id =
some_train_id
)
SELECT p.identity_id, p.name
FROM passenger p
JOIN purchases pu ON p.identity_id = pu.identity_id
JOIN ticket t ON pu.ticket_id = t.ticket_id
WHERE t.train_id = some_train_id
    AND t.carriage_id = passenger_info.carriage_id
    AND p.identity_id != some_identity_id;
```

- 订票transaction

```
CREATE PROCEDURE purchase_ticket (
    IN some_identity_id INT,
    IN departure INT,
    IN destination INT,
    IN some_seat_class CHAR(1),
    IN some_carriage_id INT,
    IN some_seat_id INT
)
BEGIN
    DECLARE some_ticket_id INT;
    START TRANSACTION;
-- 查找可用车票并锁定
    SELECT ticket_id INTO some_ticket_id
    FROM ticket t
    JOIN seat s ON t.seat_id = s.seat_id AND t.carriage_id =
s.carriage_id AND t.train_id = s.train_id
    WHERE t.available = TRUE
        AND t.depart_station = departure
        AND t.terminate_station = destination
        AND s.seat_class = some_seat_class
        AND s.carriage_id = some_carriage_id
        AND s.seat_id = some_seat_id
    FOR UPDATE;
-- 如果找不到符合条件的车票，回滚并返回错误
    IF some_ticket_id IS NULL THEN
        ROLLBACK;
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'No available ticket
found.';
    ELSE
-- 更新车票状态
        UPDATE ticket SET available = FALSE WHERE ticket_id =
some_ticket_id;
```

```
-- 插入购票记录
        INSERT INTO purchases (identity_id, ticket_id, purchase_time)
        VALUES (some_identity_id, some_ticket_id, NOW());

        COMMIT;
    END IF;
END;
```