Database Design Project Report

-Naman Patel

-Shivvrat Arya

-Group 13

- **Project Description**

Dallas Area Road Transport or DART would like one relational database to store the information about their bus transportation system to be able to carry out their work in an organized way. The DART has some major modules such as Bus, Person (Employee and Passenger) and Ticket Sales.

A Person can be an Employee or an A-class Passenger. A person can be both an employee and an A-Class passenger. Details of a person such as Person ID, Name (First, Middle, Last), Address, Gender, Date of Birth (Must be 16 years or older), and Phone number (one person can have more than one phone number) are recorded. The Person ID should have the format "PXXX" where X is a number from 0 to 9. The number of children travelling with an A-Class passenger is stored. A maximum of 5 children can travel with an A-Class Passenger.

Employee is further classified as Bus Drivers, Staff (Ticket sellers) or Ticket checkers. The start date of the employee is recorded. One bus driver can drive multiple buses and multiple drivers can drive one bus but on different dates. (At a given time in a day, only one driver drives a particular bus).

Payment information such as ID, method (cash or card), amount and other information are recorded. Ticket details such as Ticket ID, Bus ID, seat number and price are stored. The staff sells daily tickets to a person and the staff details, ticket details, person details and payment details are stored together.

An A-Star passenger is someone who has some extra privileges than an A-Class passenger. An A-Star Passenger can be an Employee or an A-Class passenger or both. Different passes are issued by DART. An A-Class passenger can buy only one pass in a month but an A-Star Passenger can buy multiple passes in a month.

Sometimes promotional discounts are offered on the passes and details such promotion ID and promotion description are recorded. The Promotional IDs are not unique and cannot be used to identify a promotion in the system.

Each A-Star Passenger is issued a travel card. The travel card details such as card ID, date of issue and other information are stored.

A-Star passengers can have guests who travel for free with them four times a month. A Guest log is maintained which stores information such as passenger ID, guest ID, guest SSN, guest name, guest address, and guest contact information. Guest IDs are temporary IDs that a person gets when they travel as a guest of an A-Star passenger. Each guest ID is not unique and cannot be used to identify a guest in the library.

Bus details such as Bus Number, License plate number, number of seats and other information are stored. Each route has many bus stops. One bus stop is part of only one route. The route and

bus stop details are stored. Each bus is parked in a terminal and the information of the terminal such as Terminal ID, Location, Date and Time are stored.

The time table information such as day and start time, end time and intervals (15 min, 20 min, 30 min) are recorded. Values for 'day' can be {M,T,W,Th,F,Sat,Sun}. A unique ID in the form of "DTXX" is given to each unique record in the timetable. For example, Day-{M,W}, StartTime-10:00, EndTime – 20:00, Interval - 15m can have ID DT01 and so on.

The information of which bus goes by which route and at what time is all stored together. The status of the bus (On Time, Delayed, or Cancelled) is recorded.

• Project Questions

1. Yes, the ability to model subclass/superclass relations is important in a transportation system like DART, because in this system, there are a lot of entities that are have similar inherent attributes with only slight changes in some attributes. So, if we don't have the option to model these entities with superclass/subclass relationships, then we not only have to repeat this information in multiple entities, but we also lose the information of these entities being related with common attribute properties. For example, in our model, we had a person entity and we had other entities like employee, A-Class passenger, A-Star passenger. If we don't model these entities as superclass and subclass relationships, it would appear as if there is no connection between these entities in the relation schema and they exist independently, which is in fact not the case here. So, modelling a superclass/subclass relationship is useful in a DART system.

2.

- Ticket validity time: Type of ticket(local or intercity), ticket validity(half day or full-day).

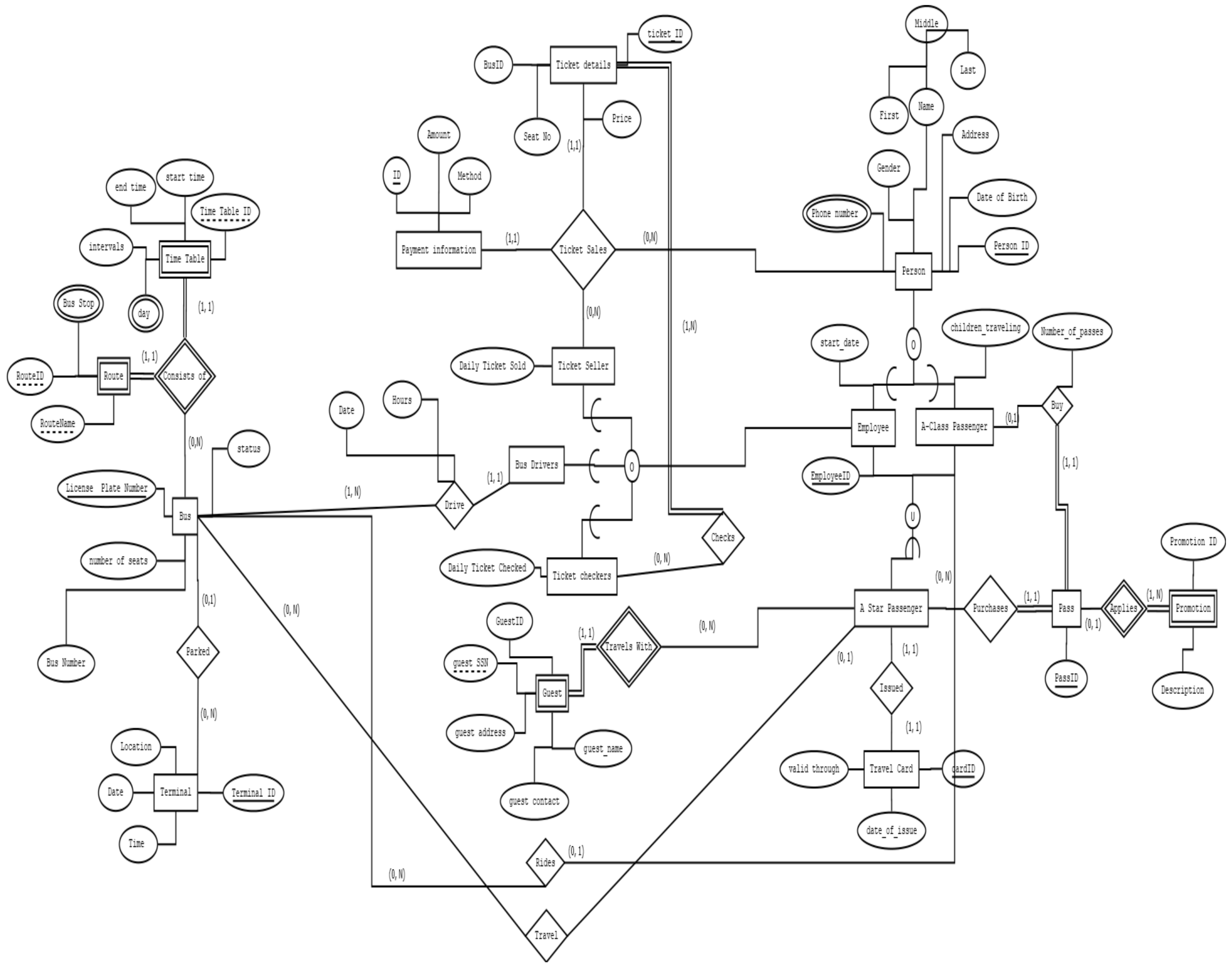
- Luggage: whether luggage can be carries, if so what are the acceptable dimensions and weight value.

- employee ratings: Employee ratings by customers on a scale of 1 to 5

- bus type: seater, sleeper, etc.

- Accidents per bus driver: No of accidents caused by a bus driver in a month/year etc.

3. The relational database offers the following advantages over other type of databases
    - The relational model structures data in a manner that avoids complexity
    - Under the relational model, accessing data in a database does not require navigating through a hierarchy of records. Users can query any table in the database, and combine related tables using special join functions to include relevant data contained in other tables in the results.
    - The relational database allows specifying constraints on the database itself as well as the individual table and tuples in order to make sure that the inherent constraints of the database systems are always followed.
    - The relational database allows for addition of more tables by adding the new information to the database in the form of tables and specifying the constraints that exist on the newly added tables.
    - Relational database models also allow for the normalization of the database tables, which ensure that there is no redundancy of data and here are fewer NULL values in the tables.

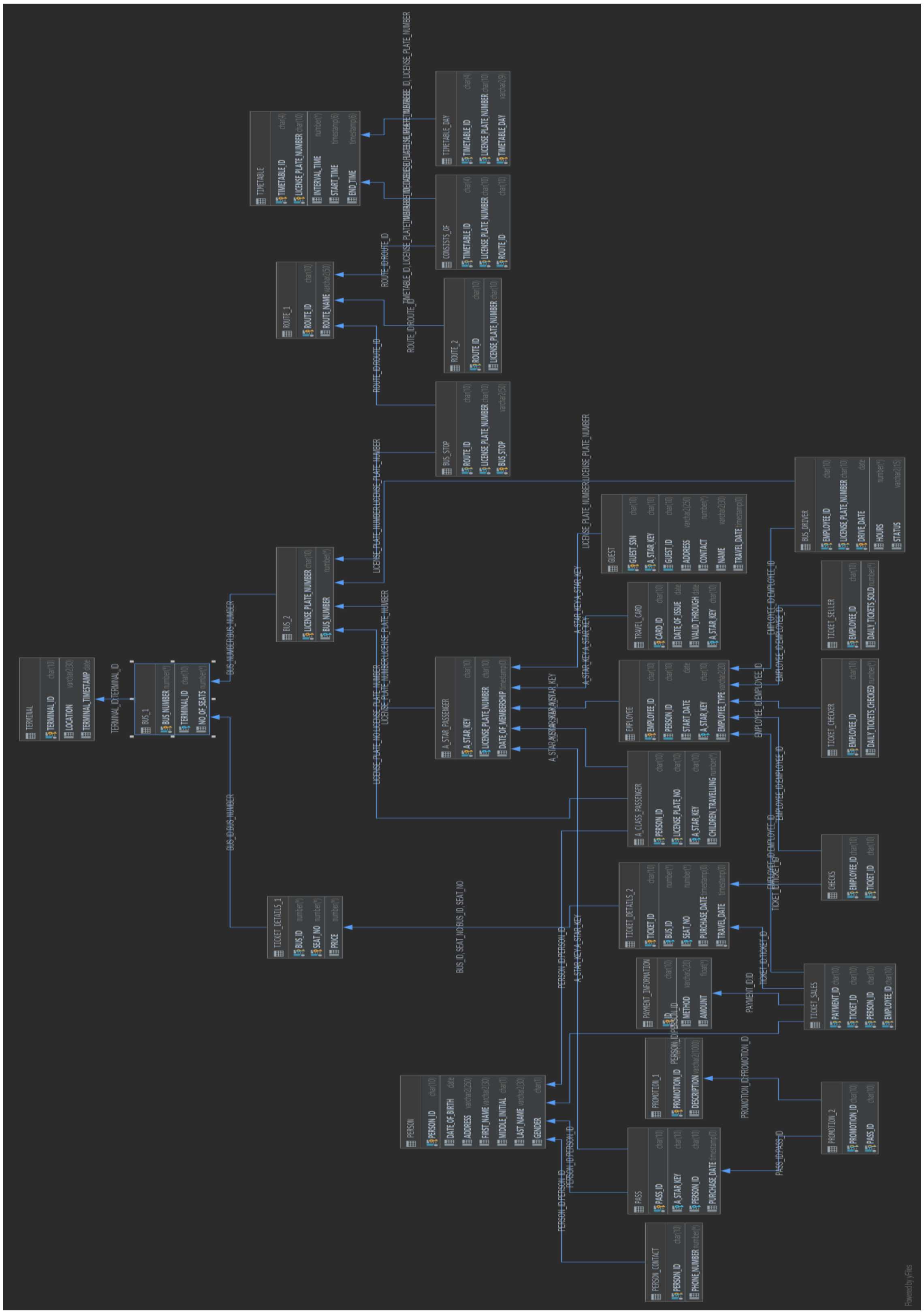

Person
Person ID
Name
First
Middle
Last
Address
Gender
Date of Birth
Phone number
O
Employee
start_date
EmployeeID
A-Class Passenger
children_traveling
Buy
Number_of_passes
(0,1)
(1,1)
Pass
PassID
Applies
(0,1)
(1,N)
Promotion
Promotion ID
Description
Purchases
(0,N)
(1,1)
U
A Star Passenger
Issued
(1,1)
(1,1)
Travel Card
cardID
valid through
date_of_issue
Travels With
(0,N)
(1,1)
Guest
GuestID
guest SSN
guest address
guest_name
guest contact
Ticket details
ticket ID
BusID
Seat No
Price
(1,1)
Ticket Sales
(1,1)
(0,N)
(0,N)
Payment information
ID
Amount
Method
Ticket Seller
Daily Ticket Sold
Bus Drivers
Ticket checkers
Daily Ticket Checked
Checks
(1,N)
(0,N)
Drive
(1,1)
(1,N)
Date
Hours
Bus
License Plate Number
number of seats
Bus Number
status
(0,N)
Consists of
(1,1)
(1,1)
Time Table
Time Table ID
start time
end time
intervals
day
Route
RouteID
RouteName
Bus Stop
Parked
(0,1)
(0,N)
Terminal
Terminal ID
Location
Date
Time
Rides
(0,1)
(0,N)
Travel
(0,N)
(0,1)

- **Assumptions for EER Diagram**

Following are the assumptions we have made for making the EER diagram:-

1. It is not necessary that a bus belongs to at least one route (it can be broken as well).
2. When the bus is not travelling, it won’t be parked to any terminal.
3. A bus can travel without any passengers.
4. Multiple ticket checkers can check a single ticket (on multiple stops)
5. Each A-star passenger is issued only one travel card.
6. Each pass can have only one type or promotion at a time, if at all.
7. A promotion can belong to multiple pass types at a time.
8. Each employee has an employee id associated with him/her
9. Ticket checker and ticket seller have attributes daily tickers checked and daily tickets sold, respectively.
10. Route Id and route name are unique for each route
11. A ticket can be sold online as well.
12. Address is stored as a simple attribute
13. Employee has overlapping specialization into bus driver, ticket seller and ticket checker
14. Pass is used as a strong entity with A-Star and A-class passengers. We are assuming each pass has a partial key of pass id
15. Each route has a rout Id.
16. Travel card has a key travel card Id.

- **Relation Schema after normalization**



VIEWS

```
1) CREATE OR REPLACE VIEW TOP_A_STAR_PASSENGER AS
   SELECT DISTINCT P.PERSON_ID, P.FIRST_NAME, P.MIDDLE_INITIAL, P.LAST_NAME,
   A_S.DATE_OF_MEMBERSHIP
   FROM PERSON P,
        A_CLASS_PASSENGER A_C,
        A_STAR_PASSENGER A_S,
        EMPLOYEE E
   WHERE ((A_S.A_STAR_KEY = A_C.A_STAR_KEY AND A_C.PERSON_ID = P.PERSON_ID)
       OR
          (A_S.A_STAR_KEY = E.A_STAR_KEY AND E.PERSON_ID = P.PERSON_ID))
     AND P.PERSON_ID IN (
       SELECT TS.PERSON_ID
       FROM A_STAR_PASSENGER A_S1,
            A_CLASS_PASSENGER A_C,
            A_STAR_PASSENGER A_S,
            EMPLOYEE E,
            TICKET_DETAILS_2 TD2,
            TICKET_SALES TS
       WHERE ((A_S.A_STAR_KEY = A_C.A_STAR_KEY)
           OR
              (A_S.A_STAR_KEY = E.A_STAR_KEY))
         AND TD2.TICKET_ID = TS.TICKET_ID
         AND (TS.PERSON_ID = E.PERSON_ID OR TS.PERSON_ID = A_C.PERSON_ID)
         AND TD2.TRAVEL_DATE > ADD_MONTHS(SYSDATE, -12)
       GROUP BY TS.PERSON_ID
       HAVING COUNT(TS.PERSON_ID) > 60);

2) CREATE OR REPLACE VIEW POPULAR_BUS AS
   SELECT B1.BUS_NUMBER, B1.TERMINAL_ID, B1.NO_OF_SEATS, B2.LICENSE_PLATE_NUMBER,
   BD.STATUS
   FROM BUS_1 B1,
        BUS_2 B2,
        BUS_DRIVER BD,
        TICKET_DETAILS_1 TD1
   WHERE B1.BUS_NUMBER = TD1.BUS_ID
     AND B1.BUS_NUMBER = B2.BUS_NUMBER
     AND BD.LICENSE_PLATE_NUMBER = B2.LICENSE_PLATE_NUMBER
     AND B1.BUS_NUMBER IN (
       SELECT BUS_NUMBER
       FROM (
                SELECT B.BUS_NUMBER, rank() OVER (ORDER BY count(*) DESC) AS rank
                FROM BUS_1 B,
                     TICKET_DETAILS_2 T
                WHERE T.BUS_ID = B.BUS_NUMBER
                  AND T.PURCHASE_DATE > ADD_MONTHS(SYSDATE, -24)
                GROUP BY B.BUS_NUMBER
            ) sub
       WHERE rank = 1
   );
```

```
3) CREATE OR REPLACE VIEW TOP_DELAYED_CANCELLED_BUS AS
   SELECT B1.BUS_NUMBER, B1.TERMINAL_ID, B1.NO_OF_SEATS, B2.LICENSE_PLATE_NUMBER,
   BD.STATUS
   FROM BUS_1 B1,
        BUS_2 B2,
        BUS_DRIVER BD
   WHERE B1.BUS_NUMBER = B2.BUS_NUMBER
     AND BD.LICENSE_PLATE_NUMBER = B2.LICENSE_PLATE_NUMBER
     AND B2.BUS_NUMBER IN (
       SELECT BUS_NUMBER
       FROM (
                SELECT B.BUS_NUMBER, rank() OVER (ORDER BY count(*) DESC) AS rank
                FROM BUS_2 B,
                     TIMETABLE T
                WHERE B.LICENSE_PLATE_NUMBER = T.LICENSE_PLATE_NUMBER
                  AND T.START_TIME BETWEEN ADD_MONTHS(SYSDATE, -1) AND SYSDATE
                GROUP BY B.BUS_NUMBER
            ) sub
       WHERE rank = 1
   );

4) CREATE OR REPLACE VIEW POTENTIAL_A_STAR_PASSENGER AS
   SELECT P.PERSON_ID, P.FIRST_NAME, P.MIDDLE_INITIAL, P.LAST_NAME,
   PC.PHONE_NUMBER
   FROM PERSON P,
        A_CLASS_PASSENGER A_C,
        PERSON_CONTACT PC
   WHERE A_C.PERSON_ID = P.PERSON_ID
     AND P.PERSON_ID = PC.PERSON_ID
     AND A_C.PERSON_ID IN (
       SELECT A_C1.PERSON_ID
       FROM A_CLASS_PASSENGER A_C1,
            TICKET_DETAILS_2 TD2,
            TICKET_SALES TS
       WHERE (
                     TS.PERSON_ID = A_C.PERSON_ID AND
                     TD2.TICKET_ID = TS.TICKET_ID AND
                     TD2.TRAVEL_DATE > ADD_MONTHS(SYSDATE, -2))
       GROUP BY A_C1.PERSON_ID
       HAVING COUNT(TD2.TRAVEL_DATE) > 40);
```

```
5) CREATE OR REPLACE VIEW TOP_EMPLOYEE AS
   SELECT P.PERSON_ID, P.FIRST_NAME, P.MIDDLE_INITIAL, P.LAST_NAME,
   PC.PHONE_NUMBER
   FROM PERSON P,
        PERSON_CONTACT PC,
        EMPLOYEE E,
        TICKET_SALES TS
   WHERE TS.PERSON_ID = E.PERSON_ID
     AND E.PERSON_ID = P.PERSON_ID
     AND P.PERSON_ID = PC.PERSON_ID
     AND P.PERSON_ID IN (
       SELECT PERSON_ID
       FROM (
                SELECT TS1.PERSON_ID, rank() OVER (ORDER BY count(*) DESC) AS
   rank
                FROM TICKET_SALES TS1,
                     TICKET_DETAILS_2 T2
                WHERE T2.PURCHASE_DATE BETWEEN ADD_MONTHS(SYSDATE, -1) AND
   SYSDATE
                GROUP BY TS1.PERSON_ID
            ) sub
       WHERE rank = 1
   );
```

QUERIES

```
1)  SELECT P.PERSON_ID, P.FIRST_NAME, P.MIDDLE_INITIAL, P.LAST_NAME,
    PC.PHONE_NUMBER, E.EMPLOYEE_TYPE
    FROM PERSON P,
         PERSON_CONTACT PC,
         EMPLOYEE E
    WHERE E.PERSON_ID = P.PERSON_ID
      AND P.PERSON_ID = PC.PERSON_ID;

2)  SELECT P.FIRST_NAME, P.MIDDLE_INITIAL, P.LAST_NAME
    FROM PERSON P,
         EMPLOYEE E,
         A_CLASS_PASSENGER AC
    WHERE AC.PERSON_ID = E.PERSON_ID
      AND P.PERSON_ID = E.PERSON_ID;
```

```
3)  SELECT AVG(count1)
    FROM (SELECT rank() OVER (ORDER BY count(TS.TICKET_ID) DESC) AS rank,
    count(TS.TICKET_ID) AS count1
          FROM TOP_A_STAR_PASSENGER TASP,
               TICKET_SALES TS,
               TICKET_DETAILS_2 TD2
          WHERE TASP.PERSON_ID = TS.PERSON_ID
            AND TD2.TICKET_ID = TS.TICKET_ID
            AND TD2.TRAVEL_DATE > ADD_MONTHS(SYSDATE, -12)
          GROUP BY TASP.PERSON_ID)
    WHERE rank < 6;
```

```
4) SELECT B2.BUS_NUMBER, R1.ROUTE_NAME
   FROM ROUTE_2 R2,
        ROUTE_1 R1,
        BUS_2 B2
   WHERE R1.ROUTE_ID = R2.ROUTE_ID
     AND B2.LICENSE_PLATE_NUMBER = R2.LICENSE_PLATE_NUMBER
     AND B2.BUS_NUMBER IN (
       SELECT BUS_NUMBER
       FROM (
                SELECT B.BUS_NUMBER, rank() OVER (ORDER BY count(*) DESC) AS rank
                FROM BUS_2 B,
                     TICKET_DETAILS_2 T
                WHERE T.BUS_ID = B.BUS_NUMBER
                GROUP BY B.BUS_NUMBER
            ) sub
       WHERE rank = 1
   );
```

```
5) SELECT B2.BUS_NUMBER
   FROM BUS_2 B2,
        BUS_DRIVER BD
   WHERE BD.LICENSE_PLATE_NUMBER = B2.LICENSE_PLATE_NUMBER
     AND BD.STATUS = ‘CANCEL’
     AND BD.DRIVE_DATE BETWEEN TRUNC(ADD_MONTHS(SYSDATE, -1), ‘MM’) AND SYSDATE
   GROUP BY B2.BUS_NUMBER
   HAVING COUNT(*) > 3;
```

```
6) SELECT T2.BUS_ID, COUNT(T2.TICKET_ID) AS BOOKINGS
   FROM TICKET_DETAILS_2 T2,
        TICKET_SALES TS
   WHERE T2.TICKET_ID = TS.TICKET_ID
   GROUP BY T2.BUS_ID;
```

```
7) SELECT P2.FIRST_NAME, P2.MIDDLE_INITIAL, P2.LAST_NAME
   FROM PERSON P2
   WHERE P2.PERSON_ID IN
         (SELECT DISTINCT P.PERSON_ID
          FROM BUS_DRIVER B,
               EMPLOYEE E,
               PERSON P
          WHERE B.EMPLOYEE_ID = E.EMPLOYEE_ID
            AND E.PERSON_ID = P.PERSON_ID
            AND B.DRIVE_DATE BETWEEN trunc(trunc(SYSDATE, ‘IW’) - 1, ‘IW’) AND
   trunc(SYSDATE, ‘IW’)
          GROUP BY P.PERSON_ID
          HAVING COUNT(*) = 7);
```

```
8) SELECT COUNT(*)
   FROM TICKET_DETAILS_2 TD2,
        TICKET_SALES TS,
        BUS_2 B2
   WHERE TD2.TICKET_ID = TS.TICKET_ID
     AND B2.BUS_NUMBER = TD2.BUS_ID
     AND B2.BUS_NUMBER IN (
       SELECT PB.BUS_NUMBER
       FROM POPULAR_BUS PB
   )
     AND TS.PERSON_ID IN
         ((SELECT P.PERSON_ID
           FROM PERSON P,
                A_CLASS_PASSENGER A_C,
                A_STAR_PASSENGER A_S,
                EMPLOYEE E
           WHERE ((A_S.A_STAR_KEY = A_C.A_STAR_KEY AND A_C.PERSON_ID =
   P.PERSON_ID)
               OR
                  (A_S.A_STAR_KEY = E.A_STAR_KEY AND E.PERSON_ID = P.PERSON_ID)))
          UNION
          (
              SELECT P.PERSON_ID
              FROM PERSON P,
                   A_CLASS_PASSENGER A_C
              WHERE (A_C.PERSON_ID = P.PERSON_ID)
          ));
```

```
9) SELECT *
   FROM TICKET_DETAILS_2 T2
   WHERE PURCHASE_DATE >
         (SELECT MAX(E.START_DATE)
          FROM EMPLOYEE E);
```

```
10) SELECT P.PERSON_ID, P.FIRST_NAME, P.MIDDLE_INITIAL, P.LAST_NAME
    FROM PERSON P,
        EMPLOYEE E,
        A_STAR_PASSENGER ASP
    WHERE P.PERSON_ID = E.PERSON_ID
     AND E.A_STAR_KEY = ASP.A_STAR_KEY
     AND ASP.DATE_OF_MEMBERSHIP BETWEEN E.START_DATE AND ADD_MONTHS(E.START_DATE,
   1);
```

```
11) SELECT R1.ROUTE_ID, R1.ROUTE_NAME
    FROM ROUTE_1 R1
    WHERE R1.ROUTE_ID IN (
       SELECT ROUTE_ID
       FROM (
                SELECT R2.ROUTE_ID, rank() OVER (ORDER BY count(*) DESC) AS rank
                FROM ROUTE_1 R2,
                     BUS_STOP BS
                WHERE R2.ROUTE_ID = BS.ROUTE_ID
                GROUP BY R2.ROUTE_ID
            ) sub
        WHERE rank = 1
    );
```

```
12) SELECT DISTINCT P.PERSON_ID, P.FIRST_NAME, P.MIDDLE_INITIAL, P.LAST_NAME
    FROM PERSON P,
        A_CLASS_PASSENGER A_C,
        A_STAR_PASSENGER A_S,
        EMPLOYEE E
    WHERE ((A_S.A_STAR_KEY = A_C.A_STAR_KEY AND A_C.PERSON_ID = P.PERSON_ID)
       OR
          (A_S.A_STAR_KEY = E.A_STAR_KEY AND E.PERSON_ID = P.PERSON_ID))
     AND A_S.DATE_OF_MEMBERSHIP < ADD_MONTHS(SYSDATE, -60);
```

```
13)  SELECT TD2.PURCHASE_DATE, TD2.BUS_ID, TD2.TICKET_ID, TD2.SEAT_NO
     FROM TICKET_DETAILS_2 TD2,
        TICKET_SALES TS,
        POTENTIAL_A_STAR_PASSENGER PASP
     WHERE TD2.TICKET_ID = TS.TICKET_ID
       AND TS.PERSON_ID = PASP.PERSON_ID
       AND TD2.PURCHASE_DATE BETWEEN ADD_MONTHS(SYSDATE, -12) AND SYSDATE;
```

TRIGGERS

```
CREATE OR REPLACE TRIGGER TRAVEL_TRIGGER
    BEFORE INSERT OR UPDATE
    ON A_CLASS_PASSENGER
    FOR EACH ROW
BEGIN
    IF :new.CHILDREN_TRAVELLING > 5
    THEN
        RAISE_APPLICATION_ERROR(-20000, 'Maximum 5 children can travel with a
particular A-Class Passenger per journey');
    end if;
end;


CREATE OR REPLACE TRIGGER A_CLASS_MONTHLY_PASS_COUNT
    BEFORE INSERT OR UPDATE
    ON PASS REFERENCING NEW AS new
    FOR EACH ROW
```

```
DECLARE
    TOTAL_PASSES number;

BEGIN
    SELECT COUNT(*)
    INTO TOTAL_PASSES
    FROM PASS P,
         A_CLASS_PASSENGER ACP
    WHERE P.PERSON_ID = ACP.PERSON_ID AND P.PERSON_ID = :new.PERSON_ID AND
TO_CHAR(:new.PURCHASE_DATE, 'YYYY-MM') = TO_CHAR(P.PURCHASE_DATE, 'YYYY-MM')
    GROUP BY ACP.PERSON_ID;
    IF TOTAL_PASSES = 1 THEN
        RAISE_APPLICATION_ERROR(-20000, 'An A class passenger can only buy one pass
per month');
    end if;
 EXCEPTION
      WHEN NO_DATA_FOUND THEN
        TOTAL_PASSES := 0;
end;



CREATE OR REPLACE TRIGGER A_STAR_MONTHLY_TOTAL_MONTHLY_GUESTS
    BEFORE INSERT OR UPDATE
    ON GUEST REFERENCING NEW AS new
    FOR EACH ROW

DECLARE
    TOTAL_GUESTS NUMBER;

BEGIN
    SELECT COUNT(*)
    INTO TOTAL_GUESTS
    FROM GUEST G
    WHERE G.A_STAR_KEY = :new.A_STAR_KEY AND TO_CHAR(:new.TRAVEL_DATE, 'YYYY-MM') =
TO_CHAR(G.TRAVEL_DATE, 'YYYY-MM')
    GROUP BY G.A_STAR_KEY;
    IF TOTAL_GUESTS >= 4 THEN
        RAISE_APPLICATION_ERROR(-20000, 'Maximum 4 guests can travel with an A-Star
Passenger per month');
    end if;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        TOTAL_GUESTS := 0;
end;
```

DDL STATEMENTS

```
create table ROUTE_1
(
   ROUTE_ID CHAR(10) not null
```

```
      constraint ROUTE_1_PK
         primary key,
   ROUTE_NAME VARCHAR2(50)
)
/

create table ROUTE_2
(
   ROUTE_ID CHAR(10) not null
      constraint ROUTE_2_PK
         primary key
      constraint ROUTE_2_ROUTE_1_ROUTE_ID_FK
         references ROUTE_1
            on delete cascade,
   LICENSE_PLATE_NUMBER CHAR(10)
)
/

create table TIMETABLE
(
   INTERVAL_TIME NUMBER
      constraint INTERVAL_VALUES
         check (INTERVAL_TIME IN (15, 20, 30)),
   START_TIME TIMESTAMP(6),
   END_TIME TIMESTAMP(6),
   TIMETABLE_ID CHAR(4) not null
      constraint ID_PATTERN_CHECK
         check (REGEXP_LIKE(TIMETABLE_ID, 'DT[0-9]{2}')),
   LICENSE_PLATE_NUMBER CHAR(10) not null,
   constraint TIMETABLE_PK
      primary key (TIMETABLE_ID, LICENSE_PLATE_NUMBER)
)
/

create table TIMETABLE_DAY
(
   TIMETABLE_ID CHAR(4) not null,
   LICENSE_PLATE_NUMBER CHAR(10) not null,
   TIMETABLE_DAY VARCHAR2(9) not null
      constraint DAY_VALUES
         check (TIMETABLE_DAY IN ('M','T','W','Th','F','Sat','Sun')),
   constraint TIMETABLE_DAY_PK
      primary key (TIMETABLE_ID, LICENSE_PLATE_NUMBER, TIMETABLE_DAY),
   constraint TIMETABLE_DAY_TIMETABLE_INTERVAL_TIME_FK
      foreign key (TIMETABLE_ID, LICENSE_PLATE_NUMBER) references TIMETABLE
         on delete cascade
)
/

create table CONSISTS_OF
(
   TIMETABLE_ID CHAR(4) not null,
   LICENSE_PLATE_NUMBER CHAR(10) not null,
   ROUTE_ID CHAR(10) not null
      constraint CONSISTS_OF_ROUTE_1__FK
```

```
         references ROUTE_1,
   constraint CONSISTS_OF_PK
      primary key (TIMETABLE_ID, LICENSE_PLATE_NUMBER, ROUTE_ID),
   constraint CONSISTS_OF_TIMETABLE___FK
      foreign key (TIMETABLE_ID, LICENSE_PLATE_NUMBER) references TIMETABLE
)
/

create table TERMINAL
(
   TERMINAL_ID CHAR(10) not null
      constraint TERMINAL_PK
         primary key,
   LOCATION VARCHAR2(30),
   TERMINAL_TIMESTAMP DATE
)
/

create table BUS_1
(
   BUS_NUMBER NUMBER not null
      constraint BUS_1_PK
         primary key,
   TERMINAL_ID CHAR(10)
      constraint BUS_1_TERMINAL__FK
         references TERMINAL,
   NO_OF_SEATS NUMBER
)
/

create table BUS_2
(
   LICENSE_PLATE_NUMBER CHAR(10) not null
      constraint BUS_2_PK
         primary key,
   BUS_NUMBER NUMBER
      constraint BUS_2_BUS_1_BUS_NUMBER_FK
         references BUS_1
            on delete cascade
)
/




create table BUS_STOP
(
   ROUTE_ID CHAR(10) not null
      constraint BUS_STOP_ROUTE_1__FK
         references ROUTE_1,
   LICENSE_PLATE_NUMBER CHAR(10) not null
      constraint BUS_STOP_BUS_2__FK
         references BUS_2,
   BUS_STOP VARCHAR2(50) not null,
   constraint BUS_STOP_PK
      primary key (ROUTE_ID, LICENSE_PLATE_NUMBER, BUS_STOP)
)
/

create table PAYMENT_INFORMATION
(
   ID CHAR(10) not null
      constraint PAYMENT_INFORMATION_1_PK
         primary key,
   METHOD VARCHAR2(20),
   AMOUNT FLOAT
)
/

create table TICKET_DETAILS_1
(
   BUS_ID NUMBER not null
      constraint TICKET_DETAILS_1_BUS_1__FK
         references BUS_1,
   SEAT_NO NUMBER not null,
   PRICE NUMBER,
   constraint TICKET_DETAILS_1_PK
      primary key (BUS_ID, SEAT_NO)
)
/

create table TICKET_DETAILS_2
(
   BUS_ID NUMBER,
   SEAT_NO NUMBER,
   TICKET_ID CHAR(10) not null
      constraint TICKET_DETAILS_2_PK
         primary key,
   PURCHASE_DATE TIMESTAMP(0),
   TRAVEL_DATE TIMESTAMP(0) not null,
   constraint TICKET_DETAILS_2_TICKET_DETAILS_1___FK
      foreign key (BUS_ID, SEAT_NO) references TICKET_DETAILS_1
)
/






create table PERSON
(
   PERSON_ID CHAR(10) not null
      constraint PERSON_PK
         primary key
      constraint ID_PATTERN
         check (REGEXP_LIKE(PERSON_ID, 'P[0-9]{3}')),
   DATE_OF_BIRTH DATE
```

```
      constraint MIN_AGE
         check ( DATE_OF_BIRTH < TO_DATE('2003-01-01 00:00:00', 'yyyy/mm/dd
hh24:mi:ss')),
   ADDRESS VARCHAR2(250),
   FIRST_NAME VARCHAR2(30),
   MIDDLE_INITIAL CHAR,
   LAST_NAME VARCHAR2(30),
   GENDER CHAR
)
/

create table PERSON_CONTACT
(
   PERSON_ID CHAR(10) not null
      constraint PERSON_CONTACT_PK
         primary key
      constraint PERSON_CONTACT_PERSON__FK
         references PERSON,
   PHONE_NUMBER NUMBER
)
/

create table A_STAR_PASSENGER
(
   A_STAR_KEY CHAR(10) not null
      constraint A_STAR_PASSENGER_PK
         primary key,
   LICENSE_PLATE_NUMBER CHAR(10)
      constraint A_STAR_PASSENGER_BUS_2__FK
         references BUS_2,
   DATE_OF_MEMBERSHIP TIMESTAMP(0)
)
/

create table EMPLOYEE
(
   PERSON_ID CHAR(10),
   EMPLOYEE_ID CHAR(10) not null
      constraint EMPLOYEE_PK
         primary key,
   START_DATE DATE,
   A_STAR_KEY CHAR(10)
      constraint EMPLOYEE_A_STAR_PASSENGER__FK
         references A_STAR_PASSENGER,
   EMPLOYEE_TYPE VARCHAR2(20)
)
/

create table BUS_DRIVER
(
   EMPLOYEE_ID CHAR(10) not null
      constraint BUS_DRIVER_1_EMPLOYEE__FK
         references EMPLOYEE,
   LICENSE_PLATE_NUMBER CHAR(10) not null
      constraint BUS_DRIVER_1_BUS_2__FK
```

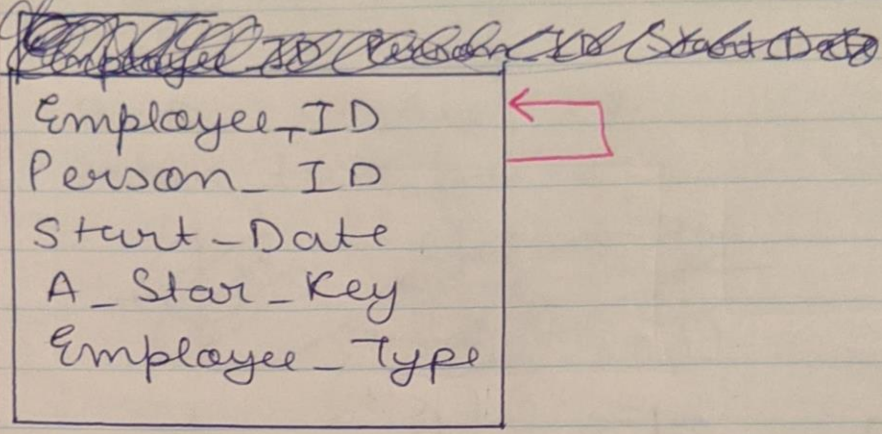
```
         references BUS_2,
   DRIVE_DATE DATE not null,
   HOURS NUMBER,
   STATUS VARCHAR2(15) default 'ON_TIME',
   constraint BUS_DRIVER_1_PK
      primary key (EMPLOYEE_ID, LICENSE_PLATE_NUMBER, DRIVE_DATE)
)
/

create table CHECKS
(
   EMPLOYEE_ID CHAR(10) not null
      constraint CHECKS_EMPLOYEE__FK
         references EMPLOYEE,
   TICKET_ID CHAR(10) not null
      constraint CHECKS_TICKET_DETAILS_2__FK
         references TICKET_DETAILS_2,
   constraint CHECKS_PK
      primary key (EMPLOYEE_ID, TICKET_ID)
)
/

create table TICKET_SELLER
(
   EMPLOYEE_ID CHAR(10) not null
      constraint TICKET_SELLER_PK
         primary key
      constraint TICKET_SELLER_EMPLOYEE__FK
         references EMPLOYEE,
   DAILY_TICKETS_SOLD NUMBER
)

create table A_CLASS_PASSENGER
(
   PERSON_ID CHAR(10) not null
      constraint A_CLASS_PASSENGER_PERSON__FK
         references PERSON,
   LICENSE_PLATE_NO CHAR(10) not null
      constraint A_CLASS_PASSENGER_BUS_2__FK
         references BUS_2,
   A_STAR_KEY CHAR(10)
      constraint A_CLASS_PASSENGER_A_STAR_PASSENGER__FK
         references A_STAR_PASSENGER,
   CHILDREN_TRAVELLING NUMBER
      constraint MAX_CHILDREN
         check ( CHILDREN_TRAVELLING < 6 ),
   constraint A_CLASS_PASSENGER_PK
      primary key (PERSON_ID, LICENSE_PLATE_NO)
)
/

create table PASS
(
   PASS_ID CHAR(10) not null
      constraint PASS_PK
```

```
         primary key,
   A_STAR_KEY CHAR(10)
      constraint PASS_A_STAR_PASSENGER__FK
         references A_STAR_PASSENGER,
   PERSON_ID CHAR(10)
      constraint PASS_EMPLOYEE__FK
         references PERSON,
   PURCHASE_DATE TIMESTAMP(0)
)
/

create table TRAVEL_CARD
(
   CARD_ID CHAR(10) not null
      constraint TRAVEL_CARD_PK
         primary key,
   DATE_OF_ISSUE DATE,
   VALID_THROUGH DATE,
   A_STAR_KEY CHAR(10)
      constraint TRAVEL_CARD_A_STAR_PASSENGER__FK
         references A_STAR_PASSENGER
)
/

create table PROMOTION_1
(
   PROMOTION_ID CHAR(10) not null
      constraint PROMOTION_1_PK
         primary key,
   DESCRIPTION VARCHAR2(1000)
)
/

create table PROMOTION_2
(
   PROMOTION_ID CHAR(10) not null
      constraint PROMOTION_2_PROMOTION_1__FK
         references PROMOTION_1
            on delete cascade,
   PASS_ID CHAR(10) not null
      constraint PROMOTION_2_PASS__FK
         references PASS,
   constraint PROMOTION_2_PK
      primary key (PROMOTION_ID, PASS_ID)
)
/

create table GUEST
(
   GUEST_SSN CHAR(10) not null,
   A_STAR_KEY CHAR(10) not null
      constraint GUEST_A_STAR_PASSENGER__FK
         references A_STAR_PASSENGER,
   GUEST_ID CHAR(10),
   ADDRESS VARCHAR2(250),
   CONTACT NUMBER,
   NAME VARCHAR2(30),
   TRAVEL_DATE TIMESTAMP(0),
   constraint GUEST_PK
      primary key (GUEST_SSN, A_STAR_KEY)
)
/
```

(We have not mentioned trivial & PK FD's)

FUNCTIONAL DEPENDENCY DIAGRAM

1) Employee :



2) Person :

Person_ID
Date_Of_Birth
Address
First_Name
Middle_Initial
Last_Name
Gender

3) Pass :

Pass ID
A_StarKey
Person_ID
Purchase_Date

4) Person Contact

Person_ID
Phone_Number

5) A-Star Passenger

A - Star Key
License_plate_no
Date_of_Membership
Travel_Date

6) Travel Card

Card_ID
Date_of_Issue
Valid_Trough
A - Star_Key

8) Promotion_2

Pass_ID
Promotion_ID

7) Promotion_1

Promotion_ID
~~Pass_ID~~
Description

9) Guest

Guest SSN
A_Star Key
Guest_ID
~~Address~~
Contact
Name

10) Ticket Sales

Payment_ID
Ticket_ID
Person_ID
Employee_ID

Payment_ID, Ticket ID
↓
Person_ID

Ticket_ID, Employee ID
↓
Person_ID

11) Consists Of

Time table ID
License Plate No
Route ID

12) Checks

Employee ID
Ticket ID

13) Terminal

Terminal_ID
Location
Terminal_TimeStamp

14) Ticket Sell

Employee_ID
Daily_Tickets_Sol

15) Payment Info I

ID
Method
Amount

~~16) Payment Info 2~~

~~ID~~
~~Amount~~

16) Ticket_details 1

Bus_ID
Seat_No
Price

17) Ticket details 2

Ticket ID
Bus ID
Seat No
Purchase Date

18) Ticket_Checker

Employee ID
Daily_Tickets_Checked

19) Bus 1

Bus_Num
TerminalID
No_of_Seats

20) Bus_2

Liscense plate #
Status
Bus_Number

21) Route_1

Route_ID
Route_Name

22) Route_2

Route_ID
License plate#

23) Bus Stop

Route_ID
License_plate_#
Bus_Stop

24) Bus_Driver

Employee ID
License plate #
Price_Date
Hours

25) Time_Table

Time Table_ID
License Plate #
Interval_Time
Start_Time
End_Time

26) TimeTable_Day

Timetable ID
License_Plate #
Timetable_Day

27) A_Class_Passenge

Person_ID
Lisenense_Plate_#
A-Star Key
Children_Travelli
Travel_Date