Member

Attribute Name	PK or AK?	Data Type & Size	Domain & Constraints	FK Reference	Description (where necessary)
member_type		Enumerated type	Not null		Child, adult or senior citizen
user_id	PK	Integer	Not null	User_Account(user_id)	

Member PSQL:

```
CREATE TYPE type_of_member AS enum('Child', 'Adult', 'Senior Citizen');
CREATE TABLE Member(
      member_type type_of_member NOT NULL,
      user_id int PRIMARY KEY REFERENCES User_Account(user_id));
```

```
bicycle_rental_facility=# create type type_of_member as enum('Child', 'Adult', 'Senior Citizen');
CREATE TYPE
bicycle_rental_facility=# CREATE TABLE Member(member_type type_of_member NOT NULL, user_id int PRIMARY KEY REFERENCES User_Account(user_id));
CREATE TABLE
```

Reasoning/References:

- I used an enum for the member type, as only these 3 types are needed. More can be added if they decide they need more types.
- I have used Table per type (TPT) inheritance for member and visitor. This is where the base class contains all the shared elements (name, address etc.) and the other classes contain only the elements that are unique to that table (in members case, type). One of the benefits of this is that it reduces data redundancy.

Visitor

Attribute Name	PK or AK?	Data Type & Size	Domain & Constraints	FK Reference	Description (where necessary)
visitor_passID	AK	serial			24 hr pass ID.
visitor_timeOfPurchase		timestamp	Not null		Date and time the visitor pass was purchased.
user_id	PK	int	NOT NULL	User_Account(user_id)	

Visitor PSQL:

```
CREATE TABLE Visitor(
     visitor_passID serial PRIMARY KEY,
     visitor_timeOfPurchase timestamp NOT NULL,
     user_id int NOT NULL REFERENCES User_Account(user_id));
```

```
bicycle_rental_facility=# CREATE TABLE Visitor(visitor_passID serial PRIMARY KEY, visitor_timeOfPurchase timestamp NOT NULL, user_id int NOT NULL REFERENCES User_Account(user_id));
CREATE TABLE
```

User Account

Attribute Name	PK or AK?	Data Type & Size	Domain & Constraints	FK Reference	Description (where necessary)
user_id	PK	Serial			
user_fullName		varchar (70)	Not null		
user_email	AK	varchar(254)	Not null		
user_phoneNum		varchar(15)	Not null		
user_address		varchar(255)	Not null		
user_postCode		varchar(12)	Can be null		

User Account PSQL:

```
CREATE TABLE User_Account(
      user_id SERIAL PRIMARY KEY,
      user_fullName varchar(70) NOT NULL,
      user_email varchar(254) NOT NULL,
      user_phoneNum varchar(15) NOT NULL,
      user_address varchar(255) NOT NULL,
      user_postCode varchar(12));
```

```
bicycle_rental_facility=# CREATE TABLE User_Account(user_id SERIAL PRIMARY KEY, user_fullName varchar(70) NOT NULL,
bicycle_rental_facility(# user_email varchar(254) NOT NULL, user_phoneNum varchar(15) NOT NULL,
bicycle_rental_facility(# user_address varchar(255) NOT NULL, user_postCode varchar(12));
CREATE TABLE
```

Reasoning/References

- After referring to the *UK Government Data Standards Catalogue*, I decided upon *70 characters for a full name* as this is what they recommend.
- For emails, *The Internet Engineering Task Force (IETF) RFC 2821* states they should be no longer than *254 characters.*
- I referred to the *international standard E.164* (the international public telecommunication numbering plan) to decide upon *15 characters* for phone numbers.
- Post code can be null as, if a visitor is filling in their details and they're from a country that doesn't use post codes, they won't be forced to enter one.
- I've decided to assume that an external company is used to safely store payment information of users. This seems like the safest way to store this kind of sensitive information and bring minimum risk to Hampshire's Social Welfare Service and its customers. After some research I've found some third-party companies that handle the encryption and storage of information, such as Authorize.net; they have a Customer Information Manager API that looks suitable.
  However, to model how this is used in my database, I've included an additional table for payment details. This will not be stored by HSWS, but will be accessed by them.

Card Information

Attribute Name	PK or AK?	Data Type & Size	Domain & Constraints	FK Reference	Description (where necessary)
card_number	PK	varchar(19)			
card_Name		varchar(70)	Can be null		Name on card
card_expiryDate		date	Not null, last_day		
card_billingAddress		varchar(255)	Can be null		
card_method		Enumerated type	Not null		Direct debit or credit/debit card
user_id		int	NOT NULL	User Account(user_id)	

Card Information PSQL:

```
CREATE TYPE payment_method AS enum('Debit Card', 'Credit Card', 'Direct Debit');
CREATE TABLE Card_Info(
      card_number varchar(19) PRIMARY KEY,
      card_Name varchar(70),
      card_expiryDate date NOT NULL,
      card_billingAddress varchar(255),
      card_method payment_method NOT NULL,
      user_id int NOT NULL REFERENCES User_Account(user_id),
      CHECK (card_expiryDate = last_day(card_expiryDate)));
```

```
bicycle_rental_facility=# CREATE TYPE payment_method AS enum('Debit Card', 'Credit Card', 'Direct Debit');
CREATE TYPE
```

```
bicycle_rental_facility=# CREATE TABLE Card_Info(
bicycle_rental_facility(# card_number varchar(19) PRIMARY KEY,
bicycle_rental_facility(# card_Name varchar(70),
bicycle_rental_facility(# card_expiryDate date NOT NULL,
bicycle_rental_facility(# card_billingAddress varchar(255),
bicycle_rental_facility(# card_method payment_method NOT NULL,
bicycle_rental_facility(# user_id int NOT NULL REFERENCES User_Account(user_id),
bicycle_rental_facility(# CHECK (card_expiryDate = last_day(card_expiryDate)));
CREATE TABLE
```

Reasoning/References

- Card numbers are in accordance with ISO/IEC 7812, which states they are no longer than 19 characters.
- For cardExpiry, I made a custom function that ensures the expiry date is set to the last day of the month. This way, the card can be used until the end of the month it expires in. It's mainly used for the check constraint, but can be used with insert and select statements if needed. It works by using the date_trunc function to truncate to month precision, then setting the date to the first of the month, adding 1 month then taking one day.
- For the billingAddress and name, I decided that if it's left null, it means the billing address/name is the same as in the user details (the case with most people) - this saves on storage.

Rental Transaction

Attribute Name	PK or AK?	Data Type & Size	Domain & Constraints	FK Reference	Description (where necessary)
transaction_id	PK	serial			
transaction_pickUpTime		timestamp	Not null		Date and time the bike is picked up from the terminal.
transaction_returnTime		timestamp	Can be null		Date and time the bike is returned.
transaction_price		decimal(6,2)	Can be null		
user_id		Int	Not null	User Account(user_id)	
bike_id		Int	Not null	Bike(bike_id)	
transaction_pickUpTerminal		Int	Not null	Terminal(terminal_id)	Which terminal the bike was taken from.
transaction_dropOffTerminal		Int	Can be null	Terminal(terminal_id)	Which terminal the bike was returned to.

Rental Transaction PSQL:

```
CREATE TABLE Rental_Transation(
      transaction_id serial PRIMARY KEY,
      transaction_pickUpTime timestamp NOT NULL,
      transaction_returnTime timestamp,
      transaction_price decimal(6,2),
      user_id int NOT NULL REFERENCES User_Account(user_id),
      bike_id int NOT NULL REFERENCES Bike(bike_id),
      transaction_pickUpTerminal int NOT NULL REFERENCES Terminal(terminal_id),
      transaction_dropOffTerminal int REFERENCES Terminal(terminal_id));
```

```
bicycle_rental_facility=# CREATE TABLE Rental_Transaction(transaction_id serial PRIMARY KEY,
bicycle_rental_facility(# transaction_pickUpTime timestamp NOT NULL, transaction_returnTime timestamp,
bicycle_rental_facility(# transaction_price decimal(6,2), user_id int NOT NULL REFERENCES User_Account(user_id), bike_id int NOT NULL REFERENCES Bike(bike_id),
bicycle_rental_facility(# transaction_pickUpTerminal int NOT NULL REFERENCES Terminal(terminal_id),
bicycle_rental_facility(# transaction_dropOffTerminal int REFERENCES Terminal(terminal_id));
CREATE TABLE
```

Reasoning/References

- returnTime, price and dropOffTerminal can all be null as they can't have a value while the bike is being used. We don't know where the bike will be returned, what time it will be returned or how long its been used (to calculate price) until the bike is back in a terminal.

Bill

Attribute Name	PK or AK?	Data Type & Size	Domain & Constraints	FK Reference	Description (where necessary)
bill_id	PK	Serial			
bill_date		date	Not null		
bill_price		decimal(6,2)	Not null		
user_id		int	Not null	User Account(user_id)	

Bill PSQL:

```
CREATE TABLE Bill(
      bill_id serial PRIMARY KEY,
      bill_date date NOT NULL,
      bill_price decimal(6,2) NOT NULL,
      user_id int NOT NULL REFERENCES User_Account(user_id));
```

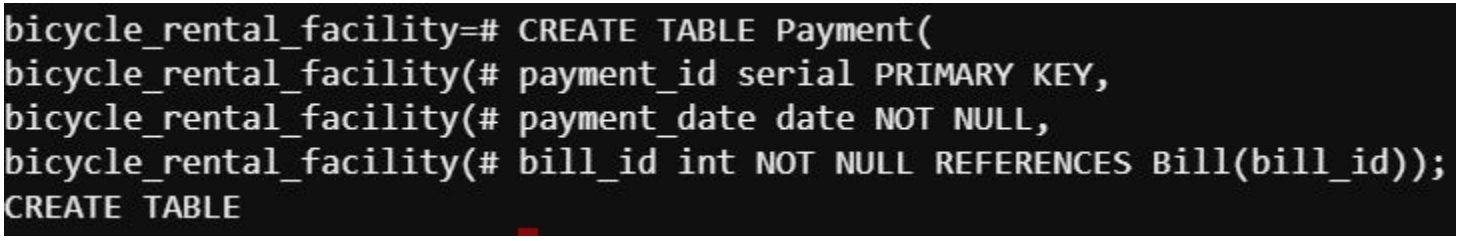
```
bicycle_rental_facility=# CREATE TABLE Bill(bill_id serial PRIMARY KEY, bill_date date NOT NULL,
bicycle_rental_facility(# bill_price decimal(6,2) NOT NULL, user_id int NOT NULL REFERENCES User_Account(user_id));
CREATE TABLE
```

Payment

Attribute Name	PK or AK?	Data Type & Size	Domain & Constraints	FK Reference	Description (where necessary)
payment_id	PK	Serial			
payment_date		date	Not null		
bill_id		int	Not null	Bill(bill_id)	

Payment PSQL:

```
CREATE TABLE Payment(
      payment_id serial PRIMARY KEY,
      payment_date date NOT NULL,
      bill_id int NOT NULL REFERENCES Bill(bill_id));
```



Terminal

Attribute Name	PK or AK?	Data Type & Size	Domain & Constraints	FK Reference	Description (where necessary)
terminal_id	PK	Serial			
terminal_name		Varchar (50)	Not null		Name set for the terminal
terminal_street		Varchar (50)	Not null		
terminal_phoneNum	AK	Varchar (15)	Not null		
terminal_maxStorage		int	Not null		
sponsor_id		int	Can be null	Sponsor(sponsor_id)	To identify the sponsor. Can be null as it's possible for there to be no sponsor.

Terminal PSQL:

```
CREATE TABLE Terminal (
      terminal_id SERIAL PRIMARY KEY,
      terminal_name varchar(50) NOT NULL,
      terminal_street varchar(50) NOT NULL,
      terminal_phoneNum varchar(15) NOT NULL,
      terminal_maxStorage int NOT NULL,
      sponsor_id int REFERENCES Sponsor(sponsor_id));
```

```
bicycle_rental_facility=# CREATE TABLE Terminal (terminal_id SERIAL PRIMARY KEY, terminal_name varchar(50) NOT NULL,
bicycle_rental_facility(# terminal_street varchar(50) NOT NULL, terminal_phoneNum varchar(15) NOT NULL, terminal_maxStorage int NOT NULL,
bicycle_rental_facility(# sponsor_id int REFERENCES Sponsor(sponsor_id));
CREATE TABLE
```

Reasoning/References

- The id for the sponsor can be null as a terminal may not always have a sponsor.

Bike

Attribute Name	PK or AK?	Data Type & Size	Domain & Constraints	FK Reference	Description (where necessary)
bike_id	PK	Serial			
bike_make		varchar (50)	Not null		
bike_model		varchar (50)	Not null		
bike_colour		varchar (50)	Not null		
bike_type		Enumerated type	Not null		Child, adult, senior citizen.
bike_latitude		decimal(8,5)	Not null		Part of the GPS tracking.
bike_longitude		decimal(8,5)	Not null		Part of the GPS tracking.
terminal_id		int	Not null	Terminal(terminal_id)	Which terminal the bike is registered at.
sponsor_id		int	Can be null	Sponsor(sponsor_id)	
contract_id		int	Not null	Contract(contract_id)	Which contract the bike is included in for the service company.

Reasoning/References

- The id for the sponsor can be null as a bike may not always have a sponsor.
- terminal_id cannot be null as, while a bike is being used it will still be registered to its last known terminal.
- The latitude and longitude are data type decimal(8,5) as, after some research, I decided to use the Decimal Degrees format as this is what most GPS devices use. 5 decimal places is a reasonable amount of precision for an object the size of a bike.

Bike PSQL:

```
CREATE TABLE Bike (
      bike_id SERIAL PRIMARY KEY,
      bike_make varchar(50) NOT NULL,
      bike_model varchar(50) NOT NULL,
      bike_colour varchar(50) NOT NULL,
      bike_type varchar(20) NOT NULL,
      bike_latitude decimal(8,5) NOT NULL,
      bike_longitude decimal(8,5) NOT NULL,
      terminal_id int NOT NULL REFERENCES Terminal(terminal_id),
      sponsor_id int REFERENCES Sponsor(sponsor_id),
      contract_id int NOT NULL REFERENCES Contract(contract_id));
```

```
bicycle_rental_facility=# CREATE TABLE Bike (bike_id SERIAL PRIMARY KEY, bike_make varchar(50) NOT NULL,
bicycle_rental_facility(# bike_model varchar(50) NOT NULL, bike_colour varchar(50) NOT NULL,
bicycle_rental_facility(# bike_type varchar(20) NOT NULL, bike_latitude decimal(8,5) NOT NULL,
bicycle_rental_facility(# bike_longitude decimal(8,5) NOT NULL, terminal_id int NOT NULL REFERENCES Terminal(terminal_id),
bicycle_rental_facility(# sponsor_id int REFERENCES Sponsor(sponsor_id), contract_id int NOT NULL REFERENCES Contract(contract_id));
CREATE TABLE
```

Sponsor

Attribute Name	PK or AK?	Data Type & Size	Domain & Constraints	FK Reference	Description (where necessary)
sponsor_id	PK	Serial			
sponsor_name		varchar(100)	Can be null		Specific product/service advertised.
sponsor_company		varchar(100)	Not null		Main company sponsoring.
sponsor_contactName		varchar(70)	Not null		Name of contact for the sponsor.
sponsor_contactPhoneNum	AK	varchar(15)	Not null		Phone number of contact for the sponsor.
sponsor_address		varchar(255)	Not null		
sponsor_ postcode		varchar(12)	Not null		
sponsor_startDate		date	Can be null		
sponsor_endDate		date	Can be null		
sponsor_fee		decimal(8,2)	Can be null		

Reasoning/References

- Sponsor_name can be null as a company may not want a specific product promoted (eg. just their logo).
- startDate, endDate and fee can all be null as a company may want their details stored for future sponsorships, even if they aren't sponsoring currently.

Sponsor PSQL:

```
CREATE TABLE Sponsor(
      sponsor_id SERIAL PRIMARY KEY,
      sponsor_name varchar(100),
      sponsor_company varchar(100) NOT NULL,
      sponsor_contactName varchar(70) NOT NULL,
      sponsor_contactPhoneNum varchar(15) NOT NULL,
      sponsor_address varchar(255) NOT NULL,
      sponsor_postCode varchar(12) NOT NULL,
      sponsor_startDate date,
      sponsor_endDate date,
      Sponsor_fee decimal(8,2));
```

```
bicycle_rental_facility=# CREATE TABLE Sponsor(sponsor_id SERIAL PRIMARY KEY, sponsor_name varchar(100),
bicycle_rental_facility(# sponsor_company varchar(100) NOT NULL, sponsor_contactName varchar(70) NOT NULL,
bicycle_rental_facility(# sponsor_phoneNum varchar(15) NOT NULL, sponsor_address varchar(255) NOT NULL,
bicycle_rental_facility(# sponsor_postCode varchar(12) NOT NULL, sponsor_startDate date, sponsor_endDate date,
bicycle_rental_facility(# sponsor_fee decimal(8,2));
CREATE TABLE
```

Contract

Attribute Name	PK or AK?	Data Type & Size	Domain & Constraints	FK Reference	Description (where necessary)
contract_id	PK	serial			
contract_fee		decimal(8,2)	Not null		

Contract PSQL:

```
CREATE TABLE Contract(
      contract_id serial PRIMARY KEY,
      contract_fee decimal(8,2) NOT NULL);
```

```
bicycle_rental_facility=# CREATE TABLE Contract(contract_id serial PRIMARY KEY, contract_fee decimal(8,2) NOT NULL);
CREATE TABLE
```

Service Company

Attribute Name	PK or AK?	Data Type & Size	Domain & Constraints	FK Reference	Description (where necessary)
service_id	PK	serial			
service_companyName		varchar(100)	Not null		
service_address		varchar(255)	Not null		
service_postcode		varchar(12)	Not null		
service_phoneNumber	AK	varchar(15)	Not null		
contract_id		int	Not null	Contract(contract_id)	

Service Company PSQL:

```
CREATE TABLE Service_Company (
      service_id serial PRIMARY KEY,
      service_companyName varchar(100) NOT NULL,
      service_address varchar(255) NOT NULL,
      service_postcode varchar(12) NOT NULL,
      service_phoneNumber varchar(15) NOT NULL,
      contract_id int REFERENCES Contract(contract_id));
```

```
bicycle_rental_facility=# CREATE TABLE Service_Company (service_id serial PRIMARY KEY, service_companyName varchar(100) NOT NULL,
bicycle_rental_facility(# service_address varchar(255) NOT NULL, service_postcode varchar(12) NOT NULL,
bicycle_rental_facility(# service_phoneNumber varchar(15) NOT NULL, contract_id int REFERENCES Contract(contract_id));
CREATE TABLE
```