Name: SAMPLE SOLUTIONS

Section (315/415/615):

MIDTERM EXAM - 601.315/415/615 - Databases

Date: October 31, 2024 in class

The total number of points in this exam is 67 for both 601.315 and 601.415/615 students, although the difficulty of questions differ. If you work at approximately 1 minute per point, you should finish on time.

Question 1 - Relational Algebra (5 points)

Given the following relations r and s:

r

Age	IdNum
30	1
31	2
62	3
77	5
19	8

s

IdNum	Dept
2	600
3	520
6	050
6	150
7	600

compute the *full outer join* (**r** $=\bowtie=$ **s**):

Age	*IdNum*	*Dept*
30	*1*	*null*
31	*2*	*600*
62	*3*	*520*
77	*5*	*null*
null	*6*	*050*
null	*6*	*150*
null	*7*	*600*
19	*8*	*null*

Question 2 - Relational Algebra (5 points)

Express the following query in Relational Algebra. The tables that are used in Questions 2-9 are found on your supplementary handout.

(a) (5 points) List the names and home zip codes of people who did **not** receive a ticket in 2024.

$Tickets2024 \leftarrow \sigma_{year=2024}(Tickets)$

$Ticketed2024 \leftarrow \Pi_{SSN}(Ticketed2024 \bowtie_{Ticketed2024.LNO=Owns.LNO} Owns)$

$NonTicketed2024 \leftarrow \Pi_{SSN}(Person) - Ticketed2024$

$Result \leftarrow \Pi_{Name,HomeZip}(Person \bowtie_{Person.SSN=NonTicketed2024.SSN} NonTicketed2024)$

Question 2 - Relational Algebra (continued)

(b) (5 points - **601.315 only**) List the names and home zip code of all people who have received tickets in all zipcodes listed in the database

$TicketedPersons \leftarrow \Pi_{SSN,TicketIssuedZip}(Owns \bowtie_{LNO} Ticket)$

$AllTicketedPersons \leftarrow TicketedPersons \div \Pi_{TicketIssuedZip}(Ticket)$

$Result \leftarrow \Pi_{Name,Homezip}(AllTicketedPersons \bowtie_{SSN} Person)$

(bb) (5 points - **601.415/615 only**) List the names and home zip code of all people who have received tickets in all zipcodes listed in the database and own every model of Red Porsche listed in the database.

$RedPorsches \leftarrow \Pi_{model}(\sigma_{Color=Red \wedge make=Porsche}(Car))$

$AllTicketed \leftarrow \Pi_{SSN}(Person \bowtie_{SSN} Owns \bowtie_{LNO} Ticket \div \Pi_{TicketIssuedZip}(Ticket))$

$AllPorsches \leftarrow \Pi_{SSN}(Person \bowtie_{SSN} Owns \bowtie_{LNO} Car \div RedPorsches)$

$Result \leftarrow \Pi_{Name,HomeZip}((AllTicketed \cap AllPorsches) \bowtie_{SSN} Person)$

(c) (5 points - **601.315 only**) List the names and home zip code of all people who have received tickets in their home zip code but not in any other zip code.

$HomeTicketed \leftarrow \sigma_{HomeZip=TicketIssuedZip}(Person \bowtie_{SSN} (Owns \bowtie_{LNO} Ticket))$

$AwayTicketed \leftarrow \sigma_{HomeZip \neq TicketIssuedZip}(Person \bowtie_{SSN} (Owns \bowtie_{LNO} Ticket))$

$Result \leftarrow \Pi_{Name,HomeZip}(Person \bowtie_{SSN} (\Pi_{SSN}(HomeTicketed - AwayTicketed)))$

(cc) (5 points - **601.415/615 only**) List the names and home zip code of all people who have received tickets in their home zip code but not in any other zip code and that same person also received a ticket in the same year that they bought any one of their cars.

$AllTicketed \leftarrow \Pi_{SSN,HomeZip}(Person \bowtie_{SSN} Owns \bowtie_{LNO} Ticket)$

$HomeTicketed \leftarrow \sigma_{HomeZip=TicketIssuedZip}(AllTicketed)$

$AwayTicketed \leftarrow \sigma_{HomeZip \neq TicketIssuedZip}(AllTicketed)$

$ResultSSNs \leftarrow \Pi_{SSN}(\sigma_{Date=Year}(Car \bowtie_{LNO} (HomeTicketed \cap AwayTicketed)))$

$Result \leftarrow \Pi_{Name,HomeZip}(ResultSSNs \bowtie_{SSN} Person)$

Question 3 - Tuple Relational Calculus (5 points)

Express the following query in Tuple Relational Calculus:

(a) (5 points - **601.315 only**) Print the license number of all red cars made in 2024 (year=2024) that are owned by someone who also received a ticket in the same zip code in which they live.

$$
\begin{aligned}
\{r \quad | \quad & \exists c \in Car \\
& \wedge \quad \exists o \in Owns \\
& \wedge \quad \exists p \in Person \\
& \wedge \quad \exists t \in Ticket \\
& \quad\quad (r.LNO = c.LNO \\
& \wedge \quad c.Color = Red \wedge c.Year = 2024 \\
& \wedge \quad c.LNO = o.LNO \\
& \wedge \quad o.SSN = p.SSN \\
& \wedge \quad p.LNO = t.LNO \wedge p.HomeZip = t.TicketIssuedZip)\}
\end{aligned}
$$

(aa) (5 points - **601.415/615 only**) Print the license number of all red cars made in 2024 (year=2024) that are owned by someone who also owns another car made before 1955 **and** who also received a ticket in the same zip code in which they live.

$$
\begin{aligned}
\{r \quad | \quad & \exists c \in Car \\
& \wedge \quad \exists o \in Owns \\
& \wedge \quad \exists o2 \in Owns \\
& \wedge \quad \exists c2 \in Car \\
& \wedge \quad \exists p \in Person \\
& \wedge \quad \exists t \in Ticket \\
& \quad\quad (r.LNO = c.LNO \\
& \wedge \quad c.Color = Red \wedge c.Year = 2024 \\
& \wedge \quad c.LNO = o.LNO \\
& \wedge \quad o.SSN = o2.SSN \\
& \wedge \quad o2.LNO = c2.LNO \wedge c2.Year < 1955 \\
& \wedge \quad o.SSN = p.SSN \\
& \wedge \quad p.LNO = t.LNO \wedge p.HomeZip = t.TicketIssuedZip)\}
\end{aligned}
$$

Question 4 - SQL (10 points)

Express the following query in SQL.

(a) (5 points - **601.315 only**) Print the total number of tickets received in each zip code in 2024.

```
SELECT TicketIssuedZip, count(TNO)
FROM Ticket
WHERE year = 2024
GROUP BY TicketIssuedZip;
```

(aa) (5 points - **601.415/615 only**) Print the total number of cars owned by each distinct Make/Model/Color combinations (e.g. Red Honda Accords as a distinct line from Rec Porsche 911's)

```
SELECT count(Car.LNO), Color, Make, Model
FROM Car, Owns
WHERE Car.LNO = Owns.LNO
GROUP BY Color, Make, Model
```

(b) (5 points - **Everyone**) Print the name and SSN of the person who has received the *most* number of parking tickets. You can assume that this person is unique.

```
SELECT Name, SSN FROM Person P,
  (SELECT X.SSN, max(X.count) as max_count
   FROM (SELECT O.SSN, count(TNO) as count
         FROM Owns, Ticket
         WHERE Owns.LNO = Ticket.LNO
         GROUP BY Owns.SSN) AS X
   GROUP BY X.SSN) AS Y
WHERE P.SSN = Y.SSN;
```

Question 5 - SQL (10 points)

Express the following queries in SQL.

(a) (5 points - **601.315 only**) How many people who owned a car manufactured before 2005 did not receive a ticket in 2024 in any car that they own (it doesn't have to be the car manufactured before 2005).

```
SELECT count(distinct P.SSN) FROM Person P, Owns O, Car C
WHERE P.SSN = X.SSN
  AND O.LNO = C.LNO
  AND C.year < 2005
  AND P.SSN NOT IN (SELECT Y.SSN
                    FROM Owns O2, Ticket T
                    WHERE O2.LNO = T.LNO
                    AND T.Year = 2024)
```

(aa) (5 points - **601.415/615 only**) How many people who owned a car manufactured after 2005 did not receive a ticket for that car in the same year that their car was manufactured.

```
SELECT count(distinct P.SSN) FROM Person P, Owns O, Car C
WHERE P.SSN = X.SSN
  AND O.LNO = C.LNO
  AND C.year > 2005
  AND P.SSN NOT IN (SELECT Y.SSN
                    FROM Owns Y, Ticket T
                    WHERE Y.LNO = T.LNO
                    AND T.Year = C.Year)
```

Question 6 - QBE (10 points)

Express the following queries in QBE. To simplify your work, table shells have been provided. Just fill in the appropriate cells with variables/values.

(a) (5 points - **601.315 only**) Print the names and phone numbers of all people who have received more tickets in 2024 than in 2023.

PERSON	SSN	Name	Phone	HomeZip
	_s	P.	P.	

CAR	LNO	Color	Make	Model	Year

OWNS	SSN	LNO
	_s	_x
	_s	_y

TICKET	TNO	LNO	TicketYear	TicketIssuedZip
	CNT.ALL._a	_x	2024	
	CNT.ALL._b	_y	2023	

Conditions
CNT.ALL._a > CNT.ALL._b

(aa) (5 points - **601.415/615 only**) Print the names and phone numbers of all people who have received more tickets in 2024 than in 2023 and own more than one car.

PERSON	SSN	Name	Phone	HomeZip
	G._ssn	P.	P.	

CAR	LNO	Color	Make	Model	Year

OWNS	SSN	LNO
	_ssn	_lno1
	_ssn	_lno2

TICKET	TNO	LNO	TicketYear	TicketIssuedZip
	CNT.UNQ.ALL._tno1	_lno1	2023	
	CNT.UNQ.ALL._tno2	_lno2	2024	

Conditions
CNT.UNQ.ALL._tno1 < CNT.UNQ.ALL._tno2

(b) (5 points - **601.315 only**) Print the names and phone numbers of all people who have received tickets in *any* 2 years in a row.

PERSON	SSN	Name	Phone	HomeZip
	_s	P.	P.	

CAR	LNO	Color	Make	Model	Year

OWNS	SSN	LNO
	_s	_x
	_s	_y

TICKET	TNO	LNO	TicketYear	TicketIssuedZip
		_x	_m	
		_y	_n	

Conditions
$\_m + 1 = \_n$

(bb) (5 points - **601.415/615 only**) Print the names and phone numbers of all people who have received tickets in *any* 3 years in a row.

PERSON	SSN	Name	Phone	HomeZip
	_ssn	P.	P.	

CAR	LNO	Color	Make	Model	Year

OWNS	SSN	LNO
	_ssn	_lno1
	_ssn	_lno2
	_ssn	_lno3

TICKET	TNO	LNO	TicketYear	TicketIssuedZip
	_tno1	_lno1	_y1	
	_tno2	_lno2	_y2	
	_tno3	_lno3	_y3	

Conditions
$\_y1 + 1 = \_y2$
$\_y2 + 1 = \_y3$

Question 7 - Understanding SQL Queries (10 points)

(a) (5 points - 601.315 only) Convert the following SQL statement to its English equivalent.

```
SELECT  Name
FROM    Person, Owns, Car
WHERE   person.SSN = owns.SSN
  AND   owns.LNO = car.LNO
  AND   car.Make = 'Honda'
  AND   car.Model = 'Accord'
  AND   car.Year = 2022;
```

Print the names of all people who own a 2022 Honda Accord.

(b) (**Everyone** - 5 points) Convert the following SQL statement to its English equivalent *(601.415/615 only).*

```
SELECT Name
FROM   Person, Owns o1, Owns o2
WHERE  Person.SSN = o1.SSN
  AND  o1.SSN = o2.SSN
  AND  o1.LNO <> o2.LNO;
```

Print the names of all people who own more than 1 car.

Question 8 - Functional Dependencies (14 points, everyone)

(a) Consider the relation **r**(A,B,C,D,E) with functional dependencies:

A → BC
CD → B
BD → E
D → A

	Circle One	List at least one process used to justify your answer
Does D → BC?	**Yes**/No	e.g. **transitivity**
Does A → C?	**Yes**/No	decomposition of A → BC
Does A → D?	Yes/**No**	
Does B → C?	Yes/**No**	
Does B → D?	Yes/**No**	
Does D → C?	**Yes**/No	transitivity (D → A, A → C) over decomposition)
Does D → E?	**Yes**/No	reflexivity and union given (D → B below, transitivity)
Does D → B?	**Yes**/No	transitivity (D → A, A → B) over decomposition)
Does CD → A?	**Yes**/No	Reflexivity (CD → D) and transitivity (with D → A)
Does CD → D?	**Yes**/No	Reflexivity
Does CD → E?	**Yes**/No	Reflexivity (CD → D) and transitivity (with D → E)

(b) Give a candidate key for R:

D is a candidate key for R because D → ABCDE (D → A given, D → D reflexivity, D → B, D → C, D → E, all shown above). Because D is a single attribute, it is by definition minimal so a candidate key (minimal superkey).

(c) Is there any other candidate key for R (if so, give one)?

NO. In particular, while CD is a superkey (essentially shown above), it is **not** a candidate key because it is not minimal, given that D alone is a candidate key (D → ABCDE).

B, C, and E don't determine anything alone (they are never on the left hand side alone). And since the only FD where A is on the left side is A → BC, it also is not a candidate key given BC doesn't determine anything (above). Thus D is the only single attribute candidate key. Also, any potential multiple-attribute candidate key would need to include D (given that nothing determines D), and since D is already a candidate key, any potential multi-attribute candidate key will not be minimal.