

Structured Query Language (SQL) Tutorial

SWEN304 / SWEN435

Trimester 1, 2024

Lecturer: Kevin Shedlock

Engineering and Computer Science



Tutorials for SWEN304 and SWEN435

Course Tutors are available at help desk labs in Room CO246. The days and times are:

- Monday, 2-3pm
- Friday 2-3pm

Outline

- SQL Jetline
- SQL Constraints:
 - CHECK constraint
 - Referential integrity constraint
`MATCH PARTIAL | MATCH FULL | MATCH SIMPLE`
- Using the same table in different context
- Correlated queries

PostgreSQL Installer 15.6 for Windows

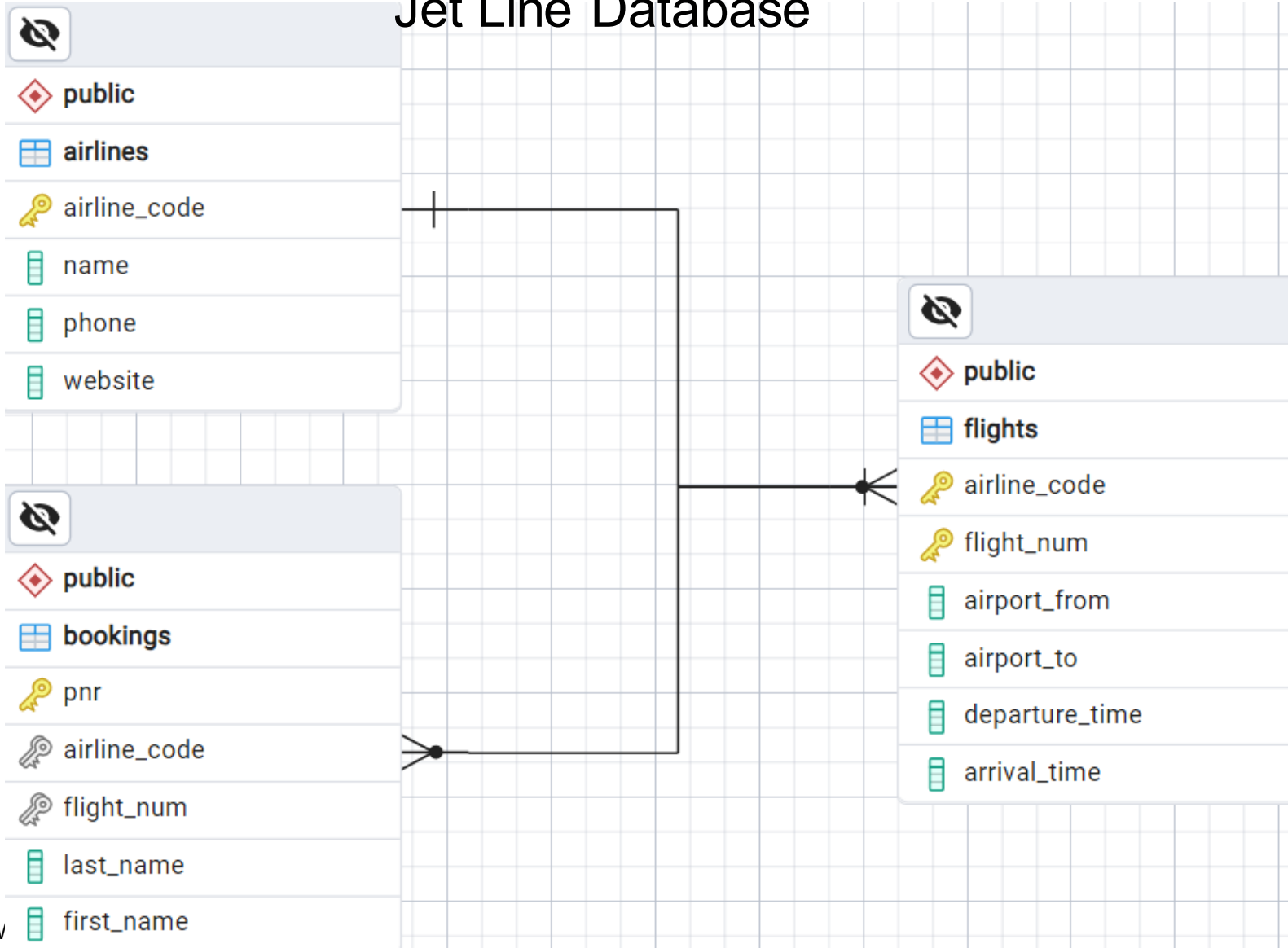
<https://www.postgresql.org/download/windows/>

PostgreSQL-Jetline Database!

- Relation of Airlines
- Relation of Flights
- Relation of Bookings

Extended Entity Relationship Diagram

Jet Line Database



Create Table airlines with four attributes airline_code; name; phone; and, website.

Assign the Primary Key to the attribute airline_code


```
create table airlines
(
  airline_code char(2) PRIMARY KEY,
  name varchar(30),
  phone varchar(15),
  website varchar(100)
);
```

Results Airlines Table

Display Jetliners database records for the airlines table

```
SELECT * FROM airlines
ORDER BY airline_code
```

```
insert into airlines
VALUES ('NZ', 'Air New Zealand', '0800 737 000', 'airnewzealand.co.nz'),
      ('JQ', 'Jetstar', null, 'jetstar.com/nz/');
```

	airline_code [PK] character 	name character varying (30) 	phone character varying (15) 	website character varying (100) 
1	JQ	Jetstar	[null]	jetstar.com/nz/
2	NZ	Air New Zealand	0800 737 000	airnewzealand.co.nz

Flights

Create table flights with five attributes airlines_code; flight_num; airport_from; airport_to; departure_time; and arrival_time.

Assign the Primary Key airline code and, a check constraint to flight_num.

```
create table flights
(
  airline_code char(2) REFERENCES airlines ON DELETE SET NULL,
  flight_num int NOT NULL
  CONSTRAINT pos_fnum CHECK (flight_num > 0),
  airport_from char(3) NOT NULL,
  airport_to char(3) NOT NULL,
  departure_time timestamp NOT NULL,
  arrival_time timestamp NOT NULL,
  CONSTRAINT flpk PRIMARY KEY (airline_code, flight_num)
);
```


Results Airlines Table

Display Jetliners database records for **flights** table

```
SELECT * FROM flights
ORDER BY airline_code, flight_num
```

```
insert into flights
VALUES ('JQ', 258, 'WLG', 'AKL', '28-03-23 12:10', '28-03-23 13:15'),
      ('NZ', 428, 'WLG', 'AKL', '28-03-23 12:55', '28-03-23 14:00'),
      ('NZ', 5562, 'CHC', 'ROT', '28-03-23 12:50', '28-03-23 14:35'),
      ('JQ', 293, 'AKL', 'ZQN', '28-03-23 13:35', '28-03-23 15:35');
```

	airline_code [PK] character	flight_num [PK] integer	airport_from character	airport_to character	departure_time timestamp without time zone	arrival_time timestamp without time zone
1	JQ	258	WLG	AKL	2023-03-28 12:10:00	2023-03-28 13:15:00
2	JQ	293	AKL	ZQN	2023-03-28 13:35:00	2023-03-28 15:35:00
3	NZ	428	WLG	AKL	2023-03-28 12:55:00	2023-03-28 14:00:00
4	NZ	5562	CHC	ROT	2023-03-28 12:50:00	2023-03-28 14:35:00

Bookings

Create table bookings with five attributes pnr; airline_code; flight_num; last_name; first_name.

Assign the Primary Key to pnr and a Foreign Key constraint with records airline_code, flight_num.

```
create table bookings
(
  pnr char(6) PRIMARY KEY,
  airline_code char(2),
  flight_num int,
  last_name varchar(12),
  first_name varchar(12),
  FOREIGN KEY (airline_code, flight_num) REFERENCES flights
);
```

Specifying CHECK Constraint

- Suppose we would like to define an additional constraint on pnr that restricts the attribute values to follow the pattern:
 - All characters are either capital letters or numbers,
 - The last character is always an 'H'.
 - How would we do this?

```
create table bookings
(
  pnr char(6) PRIMARY KEY
  CHECK constraint goes here
  airline_code char(2),
  flight_num int,
  last_name varchar(12),
  first_name varchar(12),
  FOREIGN KEY (airline_code, flight_num) REFERENCES flights
);
```

Specifying CHECK Constraint

- Suppose we would like to define an additional constraint on pnr that restricts the attribute values to follow the pattern:
 - All characters are either capital letters or numbers,
 - The last character is always an 'H'.
 - How would we do this?

```
create table bookings
(
  pnr      char(6) PRIMARY KEY
    CHECK (pnr similar to '[A-Z0-9]{5}H'),
  airline_code char(2),
  flight_num int,
  last_name varchar(12),
  first_name varchar(12),
  FOREIGN KEY (airline_code, flight_num) REFERENCES flights
);
```

Results Airlines Table

Display Jetliners database records for **bookings** table

```
SELECT * FROM bookings
ORDER BY pnr
```

```
insert into bookings
VALUES ('MFB7EH', 'NZ', '428', 'Lensen', 'Andrew'),
      ('M3A8XH', 'JQ', '293', 'Dylan', 'Bob');
```

```
insert into bookings(first_name, last_name, airline_code, flight_num, pnr)
VALUES ('jennifer', 'cooling', 'NZ', '428', 'RT3DEH');
```

	pnr [PK] character	airline_code character	flight_num integer	last_name character varying (12)	first_name character varying (12)
1	M3A8XH	JQ	293	Dylan	Bob
2	MFB7EH	NZ	428	Lensen	Andrew
3	RT3DEH	NZ	428	cooling	jennifer

Inserting some data

```
insert into airlines
```

```
VALUES ('NZ', 'Air New Zealand', '0800 737 000', 'airnewzealand.co.nz'),  
       ('JQ', 'Jetstar', null, 'jetstar.com/nz/');
```

```
insert into flights
```

```
VALUES ('JQ', 258, 'WLG', 'AKL', '28-03-23 12:10', '28-03-23 13:15'),  
       ('NZ', 428, 'WLG', 'AKL', '28-03-23 12:55', '28-03-23 14:00'),  
       ('NZ', 5562, 'CHC', 'ROT', '28-03-23 12:50', '28-03-23 14:35'),  
       ('JQ', 293, 'AKL', 'ZQN', '28-03-23 13:35', '28-03-23 15:35');
```

```
insert into bookings
```

```
VALUES ('MFB7EH', 'NZ', '428', 'Lensen', 'Andrew'),  
       ('M3A8XH', 'JQ', '293', 'Dylan', 'Bob');
```

```
insert into bookings(first_name, last_name, airline_code, flight_num, pnr)  
VALUES ('jennifer', 'cooling', 'NZ', '428', 'RT3DEH');
```

Adding constraints after declaration

```
alter table flights
  add CONSTRAINT valid_time CHECK (arrival_time > departure_time);

alter table bookings
  add CONSTRAINT capitalise_name CHECK
    ((first_name SIMILAR TO '[A-Z]*') AND
     (last_name SIMILAR TO '[A-Z]*'));
```

- Any issues?

Adding constraints after declaration

```
alter table flights  
  add CONSTRAINT valid_time CHECK (arrival_time > departure_time);
```

```
update bookings  
set first_name = UPPER(first_name),  
  last_name = UPPER(last_name);
```

```
alter table bookings  
  add CONSTRAINT capitalise_name CHECK  
  ((first_name SIMILAR TO '[A-Z]*') AND  
   (last_name SIMILAR TO '[A-Z]*'));
```


Referential Integrity – A Formal Definition

- Relations $r(N_2)$ and $r(N_1)$ satisfy referential integrity $N_2[Y] \subseteq N_1[X]$ if

$$(\forall u \in r(N_2))(\exists v \in r(N_1))(u[Y] = v[X] \vee (\exists i \in \{1, \dots, m\})(u[B_i] = \omega))$$

- Either tuples u and v are equal on X and Y values, or there exists at least one attribute in Y whose u value is null

Referential Integrity (1)

- BOOKINGS ($\{\text{pnr}, \text{airline_code}, \text{flight_num}, \text{last_name}, \text{first_name}\}, \{\text{pnr}\}$)
- FLIGHTS ($\{\text{airline_code}, \text{flight_num}, \dots\}, \{\text{airline_code} + \text{flight_num}\}$)
- How do we specify the referential integrity constraint:
BOOKINGS [$\text{airline_code}, \text{flight_num}$] \subseteq
FLIGHTS [$\text{airline_code}, \text{flight_num}$] ?

Need:

- **Referring** and **Referred** relational variables and fields
- No Match clause or Match: **FULL | SIMPLE**
- **Action:** **NO ACTION, CASCADE, SET NULL, SET DEFAULT**

Referential Integrity (2) MATCH

- MATCH clause:

SIMPLE /no MATCH clause specified (Default):

- Tuples either match on PK/FK values or the foreign key has **at least one** null-valued component

FULL:

- Tuples either match on PK/FK values or **all** foreign key components are null

NB: also PARTIAL (but not natively supported in PostgreSQL)

- *Tuples either match on PK/FK values or the non-null-valued FK components match corresponding components of at least one PK value*

Referential Integrity (3)

```
create table flights(  
  airline_code char(2) REFERENCES airlines ON DELETE SET NULL,  
  flight_num int NOT NULL  
  CONSTRAINT pos_fnum CHECK (flight_num > 0),  
  airport_from char(3) NOT NULL,  
  airport_to char(3) NOT NULL,  
  departure_time timestamp NOT NULL,  
  arrival_time timestamp NOT NULL,  
  CONSTRAINT flpk PRIMARY KEY (airline_code, flight_num));
```

```
create table bookings(  
  pnr char(6) PRIMARY KEY  
  CHECK (pnr similar to '[A-Z0-9]{5}H'),  
  airline_code char(2),  
  flight_num int,  
  last_name varchar(12),  
  first_name varchar(12),  
  FOREIGN KEY (airline_code, flight_num) REFERENCES flights  
  [MATCH <condition>] ON DELETE <action> );
```

Referential Integrity (4a)

[MATCH <condition>] ON DELETE <action>

- **No MATCH clause** (Default)
- **<action>: NO ACTION** (RESTRICT)

```
INSERT INTO BOOKINGS VALUES ('F3SKXH', 'QF', null, 'SYLES', 'JOSHUA');
```

??

```
DELETE FROM FLIGHTS WHERE airline_code = 'NZ' AND flight_num = 428;
```

??

Referential Integrity (4b – Answer)

[MATCH <condition>] ON DELETE <action>

- **No MATCH clause** (Default)
- **<action>: NO ACTION** (RESTRICT)

```
INSERT INTO BOOKINGS VALUES ('F3SKXH', 'QF', null, 'SYLES', 'JOSHUA');
```

Successful, because of MATCH default

```
DELETE FROM FLIGHTS WHERE airline_code = 'NZ' AND flight_num = 428;
```

Rejected, because of NO ACTION

Referential Integrity (5a)

[MATCH <condition>] ON DELETE <action>

- **MATCH <condition>: FULL**
- **<action>: SET NULL,**

```
INSERT INTO BOOKINGS VALUES ('F3SKXH', 'NZ', null, 'SYLES', 'JOSHUA');  
??
```

```
INSERT INTO BOOKINGS VALUES ('F3SKXH', null, null, 'SYLES', 'JOSHUA');  
?
```

```
DELETE FROM FLIGHTS WHERE airline_code = 'NZ' AND flight_num = 428;  
??
```

Referential Integrity (5b – Answer)

[MATCH <condition>] ON DELETE <action>

- **MATCH <condition>: FULL**
- **<action>: SET NULL,**

```
INSERT INTO BOOKINGS VALUES ('F3SKXH', 'NZ', null, 'SYLES', 'JOSHUA');
```

Rejected, because of MATCH FULL

```
INSERT INTO BOOKINGS VALUES ('F3SKXH', null, null, 'SYLES', 'JOSHUA');
```

Successful, because of MATCH FULL

```
DELETE FROM FLIGHTS WHERE airline_code = 'NZ' AND flight_num = 428;
```

Successful, because of SET NULL (flight tuple will be deleted, and the foreign key of the booking tuple will be null)

A Question for You (Tricky Null Value)

- What is wrong with the following query:

```
SELECT *
FROM AIRLINES
WHERE phone = Null;
```

since it returns an empty table

airline_code	name	phone	website

- Answer:
 - There is a mistake, but I don't know where
 - PostgreSQL is rubbish
 - Null is not a real value. It can be anything. So, to the questions whether

Null = Null, or

Grade = Null

PostgreSQL answers "I don't know".

Multiple uses of the same table

- SQL allows multiple occurrences of the **same** table in a **FROM** clause
- In that case, each occurrence of the same table has a **different role**, or a different context of usage
- Aliases are used to denote the context of usage

Multiple Uses of the Same Table

- **Query:** Retrieve origin and destination of flights that are longer than NZ5562 (CHC-ROT)

```
SELECT f1.airport_from, f1.airport_to
FROM flights f1, flights f2
WHERE f1.arrival_time - f1.departure_time <
      f2.arrival_time - f2.departure_time
AND f2.airline_code = 'NZ' and f2.flight_num = '5562';
```

Not an
Equi Join

- The context of **f2** is “arrival and departure time of the flight with airline_code = NZ and flight_num = 5562”
- The context of **f1** is “list of flights having longer duration than flight with airline_code = NZ and flight_num = 5562”

Anything Else?

- Any constraints you want to add?
- Any other changes?

- Joins to try?
- ...

COMPANY Database Schema

EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary	Super_ssn	Dno
-------	-------	-------	------------	-------	---------	-----	--------	-----------	-----

DEPARTMENT

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date
-------	----------------	---------	----------------

DEPT_LOCATIONS

<u>Dnumber</u>	<u>Dlocation</u>
----------------	------------------

PROJECT

Pname	<u>Pnumber</u>	Plocation	Dnum
-------	----------------	-----------	------

WORKS_ON

<u>Essn</u>	<u>Pno</u>	Hours
-------------	------------	-------

DEPENDENT

<u>Essn</u>	<u>Dependent_name</u>	Sex	Bdate	Relationship
-------------	-----------------------	-----	-------	--------------

Figure 5.5
Schema diagram for
the COMPANY
relational database
schema.

Exercises

In SQL, specify the following queries on the COMPANY database using the concept of nested queries.

- 1) Retrieve the names of all employees who work in the department that has the employee with the highest salary among all employees.
- 2) Retrieve the names of all employees whose supervisor's supervisor has '888665555' for Ssn.
- 3) Retrieve the names of employees who make at least \$10,000 more than the employee who is paid the least in the company.

Exercises

- 1) Retrieve the names of all employees who work in the department that has the employee with the highest salary among all employees.

```
SELECT LNAME
FROM EMPLOYEE
WHERE DNO = (SELECT DNO
              FROM EMPLOYEE
              WHERE SALARY = (SELECT MAX(SALARY)
                              FROM EMPLOYEE) )
```

Exercises

- 2) Retrieve the names of all employees whose supervisor's supervisor has '888665555' for Ssn.

```
SELECT LNAME
FROM EMPLOYEE
WHERE SUPERSSN IN
  (SELECT SSN
   FROM EMPLOYEE
   WHERE SUPERSSN = '888665555' )
```


- 3) Retrieve the names of employees who make at least \$10,000 more than the employee who is paid the least in the company

```
SELECT LNAME  
FROM EMPLOYEE  
WHERE SALARY >= 10000 +  
      ( SELECT MIN(SALARY)  
        FROM EMPLOYEE)
```