

Basic SQL

Presented by:

Mahdi Farnaghi

Assistant Prof.

Department of Geo-Information Processing



SQL

- Data types
- Data definition
- Data manipulation
- Data retrieval



SQL Data types

Numeric Types

Name	Storage Size	Description	Range
smallint	2 bytes	small-range integer	-32768 to +32767
integer	4 bytes	typical choice for integer	-2147483648 to +2147483647
bigint	8 bytes	large-range integer	-9223372036854775808 to 9223372036854775807
decimal	variable	user-specified precision,exact	up to 131072 digits before the decimal point; up to 16383 digits after the decimal point
numeric	variable	user-specified precision,exact	up to 131072 digits before the decimal point; up to 16383 digits after the decimal point
real	4 bytes	variable-precision,inexact	6 decimal digits precision
double precision	8 bytes	variable-precision,inexact	15 decimal digits precision
smallserial	2 bytes	small autoincrementing integer	1 to 32767
serial	4 bytes	autoincrementing integer	1 to 2147483647
bigserial	8 bytes	large autoincrementing integer	1 to 9223372036854775807



SQL Data types

Monetary Types, Character Types, and Boolean

Name	Storage Size	Description	Range
money	8 bytes	currency amount	-92233720368547758.08 to +92233720368547758.07

S. No.	Name & Description
1	character varying(n), varchar(n) variable-length with limit
2	character(n), char(n) fixed-length, blank padded
3	text variable unlimited length

Name	Storage Size	Description
boolean	1 byte	state of true or false



SQL Data types

Date/Time Types

Name	Storage Size	Description	Low Value	High Value
timestamp [(p)] [without time zone]	8 bytes	both date and time (no time zone)	4713 BC	294276 AD
TIMESTAMPTZ	8 bytes	both date and time, with time zone	4713 BC	294276 AD
date	4 bytes	date (no time of day)	4713 BC	5874897 AD
time [(p)] [without time zone]	8 bytes	time of day (no date)	00:00:00	24:00:00
time [(p)] with time zone	12 bytes	times of day only, with time zone	00:00:00+1459	24:00:00-1459
interval [fields] [(p)]	12 bytes	time interval	-178000000 years	178000000 years



SQL Data types

Other data types in PostgreSQL

- Geometric Types
- Network Address Type
- Bit String Type
- Text Search Type
- UUID Type
- XML Type
- JSON Type
- Array Type



SQL Data Definition

CREATE TABLE

```
CREATE TABLE student (  
    student_name CHAR (32),  
    student_age integer,  
    city CHAR (100),  
    PRIMARY KEY (student_name)  
);
```



Data Definition

DROP TABLE

```
DROP TABLE student;
```

ALTER TABLE

```
ALTER TABLE student ADD student_address CHAR (100);  
ALTER TABLE student DROP student_address;  
ALTER TABLE student ALTER ID CHAR(10) ;
```




Data manipulation

INSERT – Add rows of data to a table

General form:

```
INSERT INTO <table>
    [(<column1> [, <column2>]...)]
VALUES
    (<constant> [<constant>]...);
```

Example:

```
INSERT INTO student
    (student_name, student_age, city)
VALUES
    ('Fredrik', 82, 'Enschede');
```



Data manipulation

UPDATE – Change/modify existing rows in a table

General form:

```
UPDATE <table> SET  
    <column1> = <constant>  
    [, <column2> = <constant>...]  
[WHERE <condition>];
```

Example:

```
UPDATE student SET  
    student_name = 'Fredrik van ...'  
  
WHERE student_name = 'Fredrik';
```



Data manipulation

DELETE – Remove rows from a table

General form:

```
DELETE FROM <table>  
[WHERE <condition>];
```

Example:

```
DELETE FROM student  
WHERE student_name = 'Fredrik';
```



Data manipulation

Constraints are used to make sure that the integrity of the database is kept.

```
CREATE TABLE faculty (  
    faculty_id integer PRIMARY KEY,  
    faculty_name CHAR(100),  
    building_name CHAR(100)  
);
```

```
CREATE TABLE student (  
    student_name CHAR (32) NOT NULL,  
    student_age integer  
        NOT NULL CHECK(student_age >= 0) ,  
    city CHAR (100) ,  
    faculty_id integer REFERENCES faculty(faculty_id) ,  
    PRIMARY KEY (student_name)  
);
```



Data retrieval

SELECT – Retrieve rows from one (or more) table(s)

General form:

```
SELECT * | <column>[, <column>...]  
FROM <table>  
[WHERE <condition>]  
[GROUP BY <column> [HAVING <condition>]]  
[ORDER BY <column>];
```

Example:

```
SELECT *  
FROM student  
WHERE student_name = 'Fredrik';
```

Result:

Student_name	student_age	city	faculty_id
Mark	34	Enschede	5

QUESTIONS?

