Next Gen Engineering
School of Tech - INDIA

Databases - Relational

Next Gen Engineering
School of Tech - INDIA

Overview

- What is a database?
- SQL
- DDL and DML
- Using DDL SQL statements
- Using DML SQL statements
- Connecting to a database and using SQL using TypeScript

Next Gen Engineering
School of Tech - INDIA

Objectives

- Learn the features of a database
- Learn why we need databases
- Get introduced to the differences between relational and non-relational databases
- See some examples of SQL and learn about DDL and DML
- Learn, and practice using, SQL commands to interact with a database
- Learn, and practice using, TypeScript to interact with a database

Next Gen Engineering
School of Tech - INDIA

What is a Database?

A collection of structured information or data organized for convenient access.

Databases typically follow a client-server architecture:

The client might be:

- A command line tool
- GUI tool
- A library for a programming language!

Aspects of data

Table - holds a collection of columns and rows.
Column - hold different types of data.
Row - an individual entry in the table.
Constraints - automatically restrict what data can appear in a column.
Schema - the information about the structure of the database and any constraints it has.

Example table

STUDENT

studentId	firstName	lastName	birthDate	mentorId
001	Bernard	Matthews	19700101	123
002	Josh	Leeds	19410709	123
003	Vince	Levis	19930430	420

Next Gen Engineering
School of Tech - INDIA

Why do we need a database?

Storing data in memory risks data loss.
Big files are slow to read and write.
Small files adds complexity to your application.
Integrity of data is easier to maintain.
Data security is easier to manage.

Next Gen Engineering
School of Tech - INDIA

Relational vs Non-relational

What is a relation?
Two or more records that store a connection between them.
In databases - with an ID in common, like `userId`.

Next Gen Engineering
School of Tech - INDIA

Non-relational

Anything that does not rely on relations between records. Most can be broadly put into one of these categories:

- Key-value stores (like a Dictionary)
- Graph stores (like a mind-map structure)
- Column stores (as opposed to Row stores - potentially faster)
- Document stores (eg JSON files)

Next Gen Engineering
School of Tech - INDIA

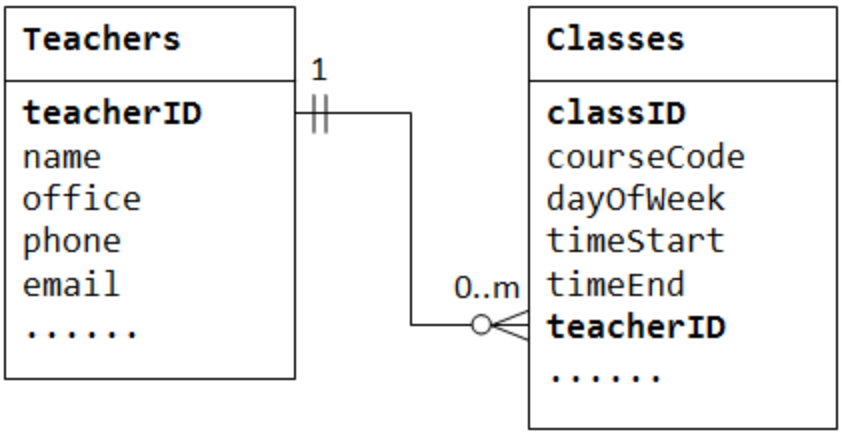
Using relations

Different kinds of relationships can exist between records:

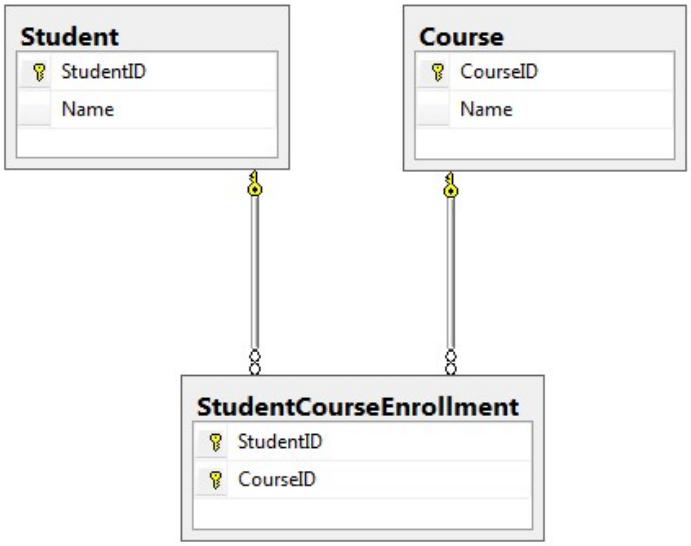
- One-to-one
- One-to-many
- Many-to-many

One to Many

- Is also Many-to-one



Many to Many



Next Gen Engineering
School of Tech - INDIA

Pros of Relational stores:

- Reduced data duplication
- Allows our database to enforce integrity for us
- Mature tooling
- How we access the data doesn't matter so much, SQL is very flexible

Cons of relational stores:

- Rigid schemas are hard to change
- Not all problems map well to relations
- Harder to scale

So, how can we utilise it?

(R)DBMS

(Relational) Database Management System.
Works with tables and the relationships between them.
Create custom Views on our data.
Program custom scripts (Stored Procedures) to query and manipulate our data.

Next Gen Engineering
School of Tech - INDIA

Some relational databases you may have heard of...

- Oracle
- MySQL
- PostgreSQL
- Microsoft SQL Server
- Microsoft Access (sort of)
- SQLite

Emoji Check:

On a high level how do you feel about the use of databases?

1. 😢 Haven't a clue, please help!
2. 😟 I'm starting to get it but need to go over some of it please
3. 😐 Ok. With a bit of help and practice, yes
4. 🙂 Yes, with team collaboration could try it
5. 😀 Yes, enough to start working on it collaboratively

Next Gen Engineering
School of Tech - INDIA

Task - 10 mins

- In groups try and come up with a design for a database for a school
- Think about what data needs storing and the relationships between that data
- Draw out your design and we'll share them shortly

Next Gen Engineering
School of Tech - INDIA

SQL

Structured Query Language (SQL)
THE language for querying relational databases.
Declarative, not imperative (i.e tell it what to do, not how to do it)

DDL vs DML

Data Definition Language
Defines how we create, alter and drop tables to work with the schema.
Data Manipulation Language
Working with the data in your tables.

DDL

Structure

```
CREATE TABLE <table_name> (
column1 <type>,
column2 <type>,
.
.
);
```

Next Gen Engineering
School of Tech - INDIA

Table Example

```
CREATE TABLE person (
    person_id INTEGER GENERATED ALWAYS AS IDENTITY,
    first_name VARCHAR(100) NOT NULL,
    surname VARCHAR(100) NOT NULL,
    age INTEGER,
    PRIMARY KEY(person_id)
);
```

- `GENERATED ALWAYS AS IDENTITY` is an auto-incrementing number
- `PRIMARY KEY` tells us what is unique for each row
- `VARCHAR` is a variable-length text field

Tables and linking data

What if we want a second table that references the first?
For example, contact info that is tied to a person?
Then we create the the second table and link back with a Constraint.
We must create the matched columns in both tables i.e. `person_id`, then link them with a `FOREIGN KEY CONSTRAINT`.

Foreign Keys example

```
CREATE TABLE contact_info (
   info_id INTEGER GENERATED ALWAYS AS IDENTITY,
   person_id INT,
   phone VARCHAR(15),
   email VARCHAR(100),
   PRIMARY KEY(info_id),
   CONSTRAINT fk_person
     FOREIGN KEY(person_id)
     REFERENCES person(person_id)
);
```

- The `CONSTRAINT` clause requires the `FOREIGN KEY` and `REFERENCES` parts too.

Next Gen Engineering
School of Tech - INDIA

ALTER TABLE

We can manipulate the structure of existing tables;

```
ALTER TABLE person ADD date_of_birth DATE;
```

Next Gen Engineering
School of Tech - INDIA

DROP TABLE

We can remove existing tables;

```
DROP TABLE contact_info;
```

(We're not expecting the use of 'DROP TABLE' often)

Emoji Check:

How do you feel about using SQL to create tables?

1. 😢 Haven't a clue, please help!
2. 😟 I'm starting to get it but need to go over some of it please
3. 😐 Ok. With a bit of help and practice, yes
4. 🙂 Yes, with team collaboration could try it
5. 😃 Yes, enough to start working on it collaboratively

Task - 5 mins

Start a postgres database running locally with Docker:

```
# Start it
docker run --name my-postgres -p 5432:5432 \
   -e POSTGRES_PASSWORD=mysecretpassword -d postgres

# Check it is running
docker ps -a
```

Next Gen Engineering
School of Tech - INDIA

Connect to your server

There are two ways to do this in general:

- Via a shell in the postgres docker container
- Install some psql tools locally and use those

> We will use the docker way

Details on following slides

Task - connect via docker

Open a shell inside the postgres container as the postgres user:

```
docker exec -it my-postgres su postgres
```

Run the `psql` tool from within the container:

```
psql
```

You should see a prompt like `psql=#` or `postgres=#`

List the databases

Some Postgres handy commands can be found on **chartio.com**
For now at our psql prompt we can:

- List the databases with `\l` or `\list` (there will only be the default ones yet)
- List your tables with the describe tables command `\dt` (there are none yet, as the default `postgres` db is empty)

Who loves a semi-colon?

> Databases do!

Statements in sql need finishing like this;
Each semicolon; Makes a separate statement;
The commands like `\dt` are postgres-only commands, not SQL, so don't need one.

Next Gen Engineering
School of Tech - INDIA

Task - groups - 15 mins

- Working in your groups create tables matching your design
- Warning: every statement must end with a `;`
- Save the commands that you create as `create_<table_name>.sql`
- Share the files with your group afterwards

Emoji Check:

How do you feel about writing SQL statements?

1. 😢 Haven't a clue, please help!
2. 😟 I'm starting to get it but need to go over some of it please
3. 😐 Ok. With a bit of help and practice, yes
4. 🙂 Yes, with team collaboration could try it
5. 😃 Yes, enough to start working on it collaboratively

School Example Teacher table

```
CREATE TABLE teacher (
   teacher_id INTEGER GENERATED ALWAYS AS IDENTITY,
   first_name VARCHAR(100) NOT NULL,
   surname VARCHAR(100) NOT NULL,
   PRIMARY KEY(teacher_id)
);
```

Next Gen Engineering
School of Tech - INDIA

School Example Course table

```
CREATE TABLE course(
   course_id INTEGER GENERATED ALWAYS AS IDENTITY,
   teacher_id INT,
   course_name VARCHAR(255),
   course_length VARCHAR(100),
   PRIMARY KEY(course_id),
   CONSTRAINT fk_person
      FOREIGN KEY(teacher_id)
     REFERENCES teacher(teacher_id)
);
```

INSERTing Data

As you would expect, this inserts (adds) a row into your table

```
INSERT INTO person (first_name, surname, age)
    VALUES ('Mike', 'Goddard', 28);
INSERT INTO person (first_name, surname, age)
    VALUES ('Emily', 'Birch', 52);
```

```
INSERT INTO contact_info (person_id, email)
    VALUES (1, 'mike@iwc.com');
INSERT INTO contact_info (person_id, email)
    VALUES (null, 'edward@iwc.com');
```

Any non-null fields are required as a value
Any field not provided with a value will default to null

INSERT with RETURNING

We can get back the id that the server created with the `RETURNING` keyword:

```
INSERT INTO person (first_name, surname)
    VALUES ('Oscar', 'Cooper') RETURNING person_id;
```

This is especially useful when writing code (like TypeScript) that is adding the data for us.

Next Gen Engineering
School of Tech - INDIA

Task - groups - 10 mins

- Working in your groups, have a go at inserting some data into your tables
- Use the `RETURNING` keyword
- Save the commands that work as `insert_<table_name>.sql` files
- Share the files with your group

Emoji Check:

How do you feel about inserting data?

1. 😢 Haven't a clue, please help!
2. 😟 I'm starting to get it but need to go over some of it please
3. 😐 Ok. With a bit of help and practice, yes
4. 🙂 Yes, with team collaboration could try it
5. 😃 Yes, enough to start working on it collaboratively

School Examples

```
INSERT INTO teacher(first_name, surname)
    VALUES ('Gatsby', 'Woodhead') RETURNING teacher_id;
INSERT INTO teacher(first_name, surname)
    VALUES ('Wiggins', 'Pickering') RETURNING teacher_id;

SELECT * FROM teacher; /* shows us the ids */

INSERT INTO course(teacher_id, course_name, course_length)
    VALUES (1, 'Physics', '3 Years') RETURNING course_id;
INSERT INTO course(teacher_id, course_name, course_length)
    VALUES (2, 'Philosophy', '4 Years') RETURNING course_id;
```

Next Gen Engineering
School of Tech - INDIA

SELECTing Data

This is how we get our data back out;
SELECT
FROM
WHERE
ORDER BY
LIMIT

SELECT

Specifies which fields to return
Takes a comma-separated list of field names

```
SELECT person_id, first_name, surname, age
```

* represents everything

```
SELECT *
```

FROM

Specifying which table you're querying against

```
SELECT person_id, first_name FROM person;
```

WHERE

Specifying a predicate that evaluates whether a row should be returned

```
SELECT * FROM person WHERE first_name = 'Mike';
```

Can take in wildcard matching

```
SELECT * FROM person WHERE first_name like '%ily%';
```

Complex WHERE

You can build where clauses that use boolean operators
You can compound the boolean operators for even more fun!

```
SELECT * FROM person
    WHERE first_name = 'Mike'
    AND (surname = 'Goddard' OR age >= 20);
```

ORDER BY

Specifying the order in which the data is returned
Optionally, the direction of the order can be appended
Can add multiple columns on which to order - this is like a primary and secondary sort (and so on)

```
SELECT * FROM contact_info
    ORDER BY email DESC, phone_number ASC;
```

LIMIT

The number of results can be limited

```
SELECT * FROM contact_info
WHERE first_name = 'Mike'
ORDER BY surname ASC
LIMIT 1;
```

Next Gen Engineering
School of Tech - INDIA

Task - groups - 5 mins

- Use `SELECT` to pull the data back out of your tables to check the `INSERT` worked correctly
- Save the commands that work as `select_<table_name>.sql` files
- Share the files with your group

Emoji Check:

How do you feel about selecting data?

1. 😢 Haven't a clue, please help!
2. 😟 I'm starting to get it but need to go over some of it please
3. 😐 Ok. With a bit of help and practice, yes
4. 🙂 Yes, with team collaboration could try it
5. 😃 Yes, enough to start working on it collaboratively

Next Gen Engineering
School of Tech - INDIA

Modifying Data

UPDATE

A way to change a row that is already stored:

```
UPDATE person SET age = 25 WHERE first_name = 'Mike';
```

Best practice is to update by the primary key id(s):

```
UPDATE person SET age = 25 WHERE person_id = 123;
```

Question: UPDATE

> What happens if we miss off the `WHERE`?

```
UPDATE person SET age = 25;
```

Answer: We change everyone at once!

DELETE

As you'd expect, delete a row given certain conditions:

```
DELETE FROM person WHERE first_name = 'Emily';
```

Best practice is to delete by the primary key id(s):

```
DELETE FROM person WHERE person_id = 456;
```

Question: DELETE

> What happens if we miss off the `WHERE`?

```
DELETE FROM person;
```

Answer: We delete everyone at once! Possibly a P45 too!

Next Gen Engineering
School of Tech - INDIA

Task - groups - 10 mins

- Use UPDATE to update a row
- Use DELETE to delete a row
- Save the commands that work as `<command>_<table_name>.sql` files
- Share the files with your group

Emoji Check:

How do you feel about updating and deleting data?

1. 😢 Haven't a clue, please help!
2. 😟 I'm starting to get it but need to go over some of it please
3. 😐 Ok. With a bit of help and practice, yes
4. 🙂 Yes, with team collaboration could try it
5. 😃 Yes, enough to start working on it collaboratively

Next Gen Engineering
School of Tech - INDIA

Discussion - 2 mins

Did anyone run into problems?

Referential Integrity

If you try and delete a row that is linked by a foreign key Postgres will complain. It's enforcing data integrity for us!

```
DELETE FROM person WHERE person_id = 1;

ERROR: update or delete on table "person" violates
    foreign key constraint "fk_person" on table
"contact_info"
DETAIL:  Key (person_id)=(1) is still referenced from
    table "contact_info".
```

Cascade deletes

These tell Postgres we want to delete child records if we delete the parent.
We can't alter constraints so we will need to delete the existing constraint first and then add a new one that will cascade deletes:

```
ALTER TABLE contact_info DROP CONSTRAINT fk_person;

ALTER TABLE contact_info
ADD CONSTRAINT fk_person
FOREIGN KEY(person_id)
REFERENCES person(person_id)
ON DELETE CASCADE;

DELETE FROM person WHERE person_id = 1;
```

Emoji Check:

How do you feel about concept of referential integrity and cascading deletes?

1. 😢 Haven't a clue, please help!
2. 😟 I'm starting to get it but need to go over some of it please
3. 😐 Ok. With a bit of help and practice, yes
4. 🙂 Yes, with team collaboration could try it
5. 😀 Yes, enough to start working on it collaboratively

JOINS

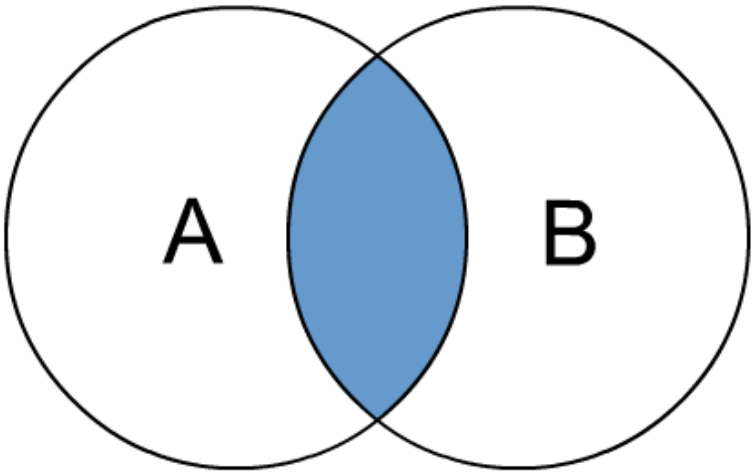
As you'd expect, we can join two or more tables in our query to get combined results.

Example

```sql
SELECT p.first_name, c.email
FROM person p
JOIN contact_info c ON p.person_id = c.person_id;
```

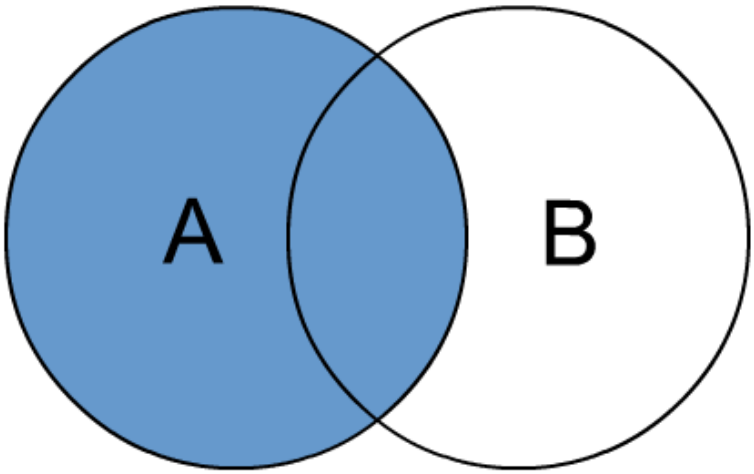
INNER JOIN

Rows matching the condition of the join will be returned:



LEFT OUTER JOIN

Returns the matching data of A and B, as well as everything from A:



What will this return?

Given some of the sample data we saw on previous slides, what does this give us?

```
SELECT p.first_name, c.email
FROM person p
LEFT JOIN contact_info c ON p.person_id = c.person_id;
```

It gives us the full set of people with blanks where the contacts don't match, and no Edward as he only has a Contact info:

```
 first_name | email
------------+-----------------
 Mike       | mike@iwc.com
 Emily      |
```

How about this?

Given some of the sample data we saw on previous slides, what does this give us?

```
SELECT p.first_name, c.email
FROM person p
RIGHT JOIN contact_info c ON p.person_id = c.person_id;
```

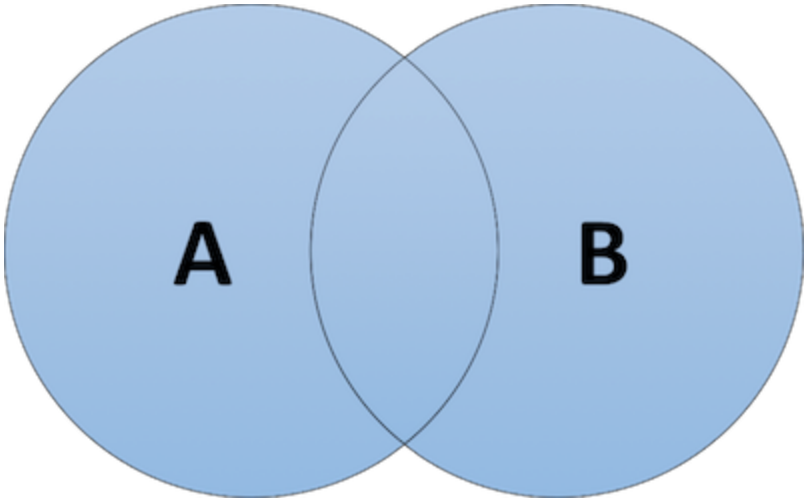
RIGHT OUTER JOIN

It gives us everything from the RHS (contact_info) with blanks from missing People (so no Emily):

```
 first_name | email
------------+-----------------
 Mike       | mike@iwc.com
            | edward@iwc.com
```

FULL OUTER JOIN

Returns everything from both tables, regardless of if they match:



Given some of the sample data we saw on previous slides, what does this give us?

```
SELECT p.first_name, c.email
FROM person p
FULL OUTER JOIN contact_info c ON p.person_id = c.person_id;
```

It gives us all the rows with blanks inserted where there are no matches:

```
 first_name | email
------------+-----------------
 Mike       | mike@iwc.com
            | edward@iwc.com
 Emily      |
```

Emoji Check:

How do you feel about joining table data?

1. 😢 Haven't a clue, please help!
2. 😟 I'm starting to get it but need to go over some of it please
3. 😐 Ok. With a bit of help and practice, yes
4. 🙂 Yes, with team collaboration could try it
5. 😃 Yes, enough to start working on it collaboratively

Next Gen Engineering
School of Tech - INDIA

Task - 5 mins

- Use some JOINs to get data out of your database from more than one table at a time
- Save the commands that work as `join_<table_name>.sql` files
- Share the files with your group

Emoji Check:

How do you feel about writing JOIN statements in SQL?

1. 😢 Haven't a clue, please help!
2. 😟 I'm starting to get it but need to go over some of it please
3. 😐 Ok. With a bit of help and practice, yes
4. 🙂 Yes, with team collaboration could try it
5. 😃 Yes, enough to start working on it collaboratively

Next Gen Engineering
School of Tech - INDIA

Functions

Many built-in functions exist for each RDBMS you're using. Some useful functions you may want to use:

- `SUBSTRING`
- `AVERAGE`, `MIN`, `MAX`
- `CURRENT_DATE`
- `COUNT`
- `UPPER`
- `LOWER`

Some examples

```
SELECT substring(first_name, 1, 3) from person;

SELECT substring(first_name from 1 for 3) from person;
```

Gives us:

```
 substring
-----------
 Mik
 Emi
```

Next Gen Engineering
School of Tech - INDIA

Task - groups - 5 mins

- Use SUBSTRING in a query
- Use UPPER or LOWER in the same query as the SUBSTRING

Emoji Check:

How do you feel about functions in SQL?

1. 😢 Haven't a clue, please help!
2. 😟 I'm starting to get it but need to go over some of it please
3. 😐 Ok. With a bit of help and practice, yes
4. 🙂 Yes, with team collaboration could try it
5. 😃 Yes, enough to start working on it collaboratively

Next Gen Engineering
School of Tech - INDIA

Exercises: Please get the files now

Everyone needs to download the `cheatsheets`, `exercises` and `solutions` zips for distribution
Please all do that now

Troubleshooting

On Windows (WSL/Ubuntu), but also some Macs/Unix, you may need to make the `*.sh` utility script files executable after unzipping;

```
cd exercises/relational-express-postgres
chmod -R a+x *.sh
```

Next Gen Engineering
School of Tech - INDIA

Connecting to the Database through TypeScript

We will live code this step by step.
You should have the `exercises.zip` from the instructor.

- Make a new folder
- Extract the zip file
- You should find the `database-helper.ts` file is a shell
- ...Because we will add the required code in this session

Docker to the rescue

We will use Docker Compose to set up a database, all the data, and a skeleton api server.
This means we all start from the same place (which is the tables and data from these slides).
Then, we can do the fun bit of coding the TypeScript together.

Useful scripts for Exercises

Some utility scripts are provided so we can concentrate on the required TypeScript:

- `./docker-compose-run-all.sh` to build and test everything
- `./docker-compose-setup-all.sh` to build and start Postgres and the api app
- `./call-express-app.sh` to test your API code
- `./docker-compose-retest-app.sh` to re-build and test the api app

Don't run any yet!

Next Gen Engineering
School of Tech - INDIA

Task: Quick-ish database setup.

To make sure we all have the same db loaded,

- Change to the `exercises/relational-express-postgres` folder
- Run script `./docker-compose-run-all.sh`

This:

- Makes a new postgres server with tables and data,
- Brings up an skeleton api server
- Tests the api server methods

Next Gen Engineering
School of Tech - INDIA

Demo: Compose file

Now we all have the same db content loaded, lets see what is in there:
Look at this file in `exercises/relational-express-postgres`:

- `docker-compose.yml`

Don't run compose directly, the script has done it for us!

Next Gen Engineering
School of Tech - INDIA

Demo: Setup scripts

Look at these files in `exercises/relational-express-postgres`:

- `db-scripts/01-setup-tables.sh`
- `db-scripts/02-setup-data.sh`
- `db-scripts/03-select-data.sh`

Don't run these files, compose has done it for us!

Next Gen Engineering
School of Tech - INDIA

Demo: Peek inside the schema

To make sure we all have the same schema loaded,

- lets looks at `db-scripts/01-setup-tables.sh`
- This puts tables in the default `postgres` db

Don't run this file, compose has done it for us!

Next Gen Engineering
School of Tech - INDIA

Demo: Peek inside the data.

To make sure we all have the same data loaded,

- lets looks at `db-scripts/02-setup-data.sh`
- This puts data in the tables

Don't run this file, compose has done it for us!

Demo: Quick-ish data check.

To make sure we all have the same data loaded,

- lets looks at `db-scripts/03-select-data.sh`
- This selects data from the tables

Don't run this file, compose has done it for us!

Emoji Check:

How do you feel about the data that has been setup in our app?

1. 😢 Haven't a clue, please help!
2. 😟 I'm starting to get it but need to go over some of it please
3. 😐 Ok. With a bit of help and practice, yes
4. 🙂 Yes, with team collaboration could try it
5. 😀 Yes, enough to start working on it collaboratively

Connecting

The first step is to create a database client, then connect with it.
But, a single client connection used by multiple users on an API will execute queries one at a time... ouch!
We could make lots of connections when we need them but... Handshaking new connections takes time.
Your DB server will only support so many concurrent connections before it falls over!

Next Gen Engineering
School of Tech - INDIA

Connection pools

Connection pools let us have a bunch of connections ready to re-use.
We can have, for example, 10 connections ready to go, so that our API won't choke for lots of users.

See how `database-helper.ts` has this:

```
const { Pool } = require("pg")
const pool = new Pool({
  user: "postgres",
  host: "my-postgres",
  database: "",
  password: "mysecretpassword",
  port: 5432,
  max: 10,
  idleTimeoutMillis: 60000,
  connectionTimeoutMillis: 10000,
})
```

The `host` value will vary between docker (`my-postgres`) and running node locally (`localhost`)... Why?

Next Gen Engineering
School of Tech - INDIA

PG library for TS

The library we are using has a useful method `pool.query()`.
This takes some sql to run, optional parameters, and a callback function for when it is complete.
We'll use this on the following slides and exercises.

This sample code would get some data if we ran it:

```
const results = await pool.query(
  "SELECT * FROM table_name ORDER BY some_id ASC;"
)
console.log(results.rows)
```

However we want this in an express app...

Code-along exercise

> However we want this in an express app.

- We'll now start live coding some methods in our express app
- We need to extend the `database-helper.ts` file

Code-along: SELECT all (15 mins)

> Complete the `getTeachers` method in `database-helper` file using the previous example code.

> Redeploy your app (use one of the useful scripts mentioned before)

> Call the api (use one of the useful scripts mentioned before)

Next Gen Engineering
School of Tech - INDIA

Emoji Check:

How do you feel about implementing selecting all teachers?

1. 😢 Haven't a clue, please help!
2. 😟 I'm starting to get it but need to go over some of it please
3. 😐 Ok. With a bit of help and practice, yes
4. 🙂 Yes, with team collaboration could try it
5. 😀 Yes, enough to start working on it collaboratively

Parameterised queries

The `query()` function takes an optional ordered list of arguments, like so:

```
const results = pool.query(
  "SELECT * FROM table_name WHERE my_id = $1;",
  [my_id]
)
console.log(results.rows)
```

You can have many i.e. `$1`, `$2`, ...`$10`.

Code-along: SELECT by id

> Complete the `getTeacherById` method in `database-helper`

> Redeploy your app (use one of the useful scripts mentioned before)

> Call the api (use one of the useful scripts mentioned before)

Emoji Check:

How do you feel about implementing select by ID?

1. 😢 Haven't a clue, please help!
2. 😟 I'm starting to get it but need to go over some of it please
3. 😐 Ok. With a bit of help and practice, yes
4. 🙂 Yes, with team collaboration could try it
5. 😃 Yes, enough to start working on it collaboratively

Code-along: INSERT

> Complete the `createTeacher` method in `database-helper`. Hint: in `server.ts` this is a `POST`.

> Redeploy your app (use one of the useful scripts mentioned before)

> Call the api (use one of the useful scripts mentioned before)

Next Gen Engineering
School of Tech - INDIA

Emoji Check:

How do you feel about inserting data?

1. 😢 Haven't a clue, please help!
2. 😟 I'm starting to get it but need to go over some of it please
3. 😐 Ok. With a bit of help and practice, yes
4. 🙂 Yes, with team collaboration could try it
5. 😃 Yes, enough to start working on it collaboratively

Code-along: UPDATE

> Complete the `updateTeacher` method in `database-helper`. Hint: in `server.ts` this is a `PUT`.

> Redeploy your app (use one of the useful scripts mentioned before)

> Call the api (use one of the useful scripts mentioned before)

Next Gen Engineering
School of Tech - INDIA

Emoji Check:

How do you feel about updating data?

1. 😢 Haven't a clue, please help!
2. 😟 I'm starting to get it but need to go over some of it please
3. 😐 Ok. With a bit of help and practice, yes
4. 🙂 Yes, with team collaboration could try it
5. 😃 Yes, enough to start working on it collaboratively

Code along - DELETE

> Complete the `deleteTeacher` method in `database-helper`. Hint: in `server.ts` this is a `DELETE`.

> Redeploy your app (use one of the useful scripts mentioned before)

> Call the api (use one of the useful scripts mentioned before)

Emoji Check:

How do you feel about deleting data?

1. 😢 Haven't a clue, please help!
2. 😟 I'm starting to get it but need to go over some of it please
3. 😐 Ok. With a bit of help and practice, yes
4. 🙂 Yes, with team collaboration could try it
5. 😃 Yes, enough to start working on it collaboratively

Next Gen Engineering
School of Tech - INDIA

Overview - recap

- What is a database?
- What is SQL?
- What are DML and DDL?
- Which SQL commands can you remember and what do they do?
- Connecting with TypeScript

Next Gen Engineering
School of Tech - INDIA

Objectives - recap

- Learnt about the features of a database
- Learnt why we need databases
- Got introduced to the differences between relational and non-relational
- Saw some examples of SQL and learn about DDL and DML
- Learnt, and practiced using, SQL commands to interact with a database
- Learnt, and practiced using, TypeScript to interact with a database

Next Gen Engineering
School of Tech - INDIA

Emoji Check:

On a high level, do you think you understand the main concepts of this session? Say so if not!

1. 😢 Haven't a clue, please help!
2. 😟 I'm starting to get it but need to go over some of it please
3. 😐 Ok. With a bit of help and practice, yes
4. 🙂 Yes, with team collaboration could try it
5. 😀 Yes, enough to start working on it collaboratively

Speaker notes