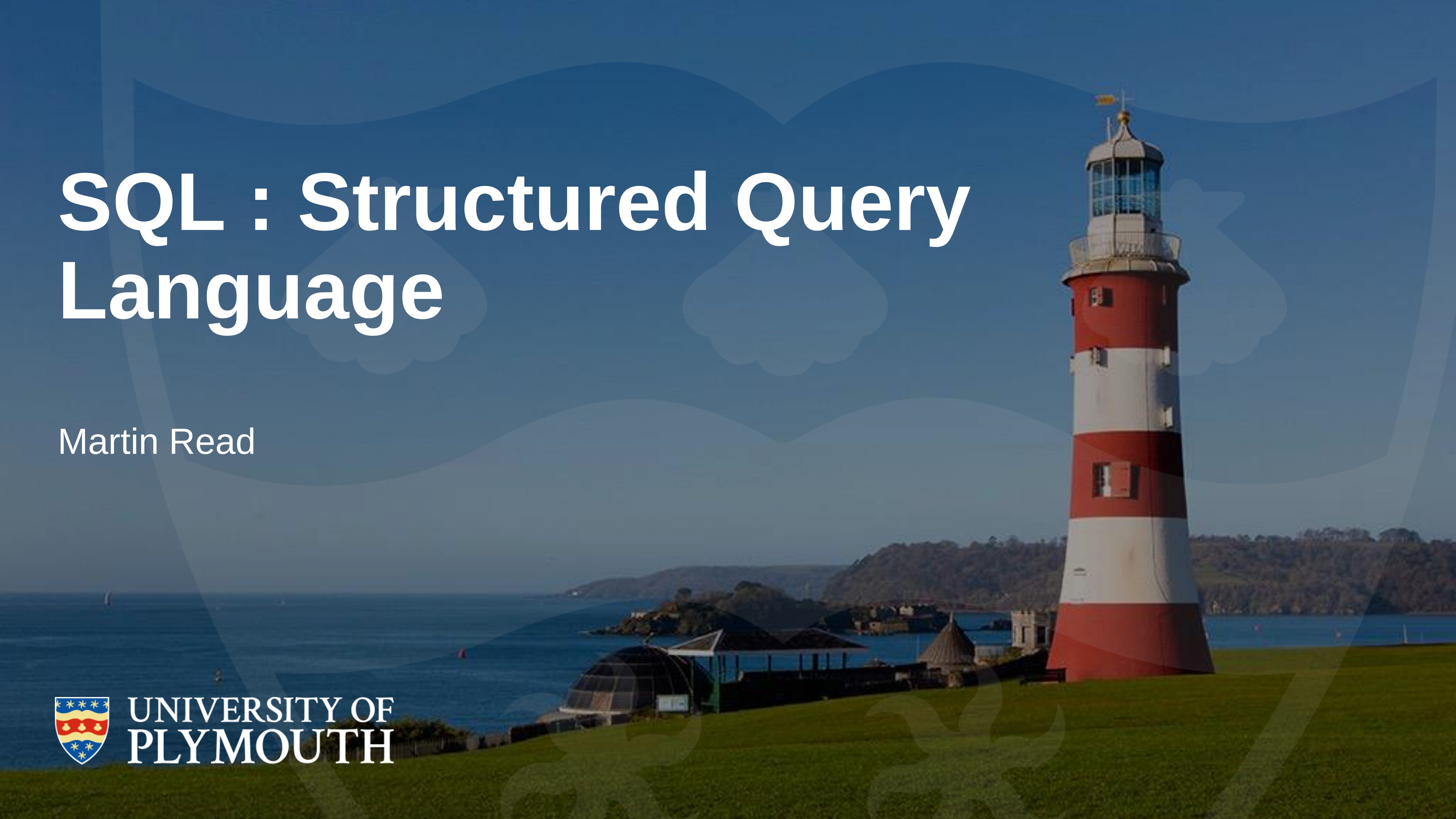

SQL : Structured Query Language

Martin Read

UNIVERSITY OF PLYMOUTH

This Lecture

Aim:

- Introduce Structured Query Language (SQL)

Learning Outcomes:

- Write SQL statements that will build up a set of tables within an RDBMS implementation
- Write SQL statements that will manipulate the existing structures to add, modify and remove columns and constraints.

What is Structured Query Language?

- Originally developed by IBM for querying, altering and defining relational databases
- A database computer language designed for managing data in relational database management systems (RDBMS)
- A **declarative language**, though now also includes procedural elements

SQL comprises?

- DDL – Data Definition Language
- DML – Data Manipulation Language
- DCL – Data Control Language

- **Applies to any aspect of RDBs**
  - can create/delete databases, tables, fields
  - can insert/update/delete/query data
  - can define access controls

Oracle

Data Retrieval	SELECT

Data Manipulation Language	INSERT
	UPDATE
	DELETE

Data Definition Language	CREATE ALTER DROP RENAME TRUNCATE
Transaction Control	COMMIT ROLLBACK SAVEPOINT
Data Control Language	GRANT REVOKE

SQL SERVER

Data Retrieval	SELECT

Data Manipulation Language	INSERT
	UPDATE
	DELETE

Data Definition Language	CREATE ALTER DROP TRUNCATE TABLE
Transaction Control	COMMIT ROLLBACK SAVE TRAN
Data Control Language	GRANT REVOKE

Data Definition Language (DDL)

- **DDL** is a syntax for creating & modifying database objects such as tables, indices & users
- **DDL** contains far more statements than we can present here,
  - & each statement is far more complex than we show in this introduction
  - If you want to master this material, you will need to go through the SQL Server documentation
- **DDL statements are used to build & modify the structure of your tables** & other objects in the database
- When you execute a DDL statement, it takes effect immediately

Components/syntax of SQL

Reserved words	shown in upper case; e.g., SELECT
User-defined words	shown in lower case; e.g., customer_number
| (vertical bar)	indicates a selection; e.g., a | b | c (a or b or c)
{ } (braces)	indicate a required element; e.g., {a}
[ ] (square brackets)	indicate an optional element; e.g., [b]
… (ellipsis)	indicates optional repetition; e.g., {a | b} [,c…]

Lay any commands out neatly for legibility

https://docs.microsoft.com/en-us/sql/t-sql/language-elements/transact-sql-syntax-conventions-transact-sql?view=sql-server-ver15

Components/syntax of SQL

- Some versions of SQL are case sensitive

  Case sensitivity only exists in literal character strings, thus

  ‘smith’ ‘SMITH’ ‘Smith’

  are each different
- Oracle is case sensitive, SQL Server is NOT case sensitive

System Datatypes in SQL Server

- **Numeric**

  int bigint smallint tinyint

  numeric bit decimal money

  float real

- **Date and time**

  date datetime time

Character strings

  char varchar

- **Binary strings**

  binary image

Data Definition Language (DDL) commands

- CREATE creates an object (e.g. a table) in the database
- ALTER modifies the structure of an existing object in various ways – e.g. adding a column to an existing table
- DROP deletes an object in the database, usually irretrievably

CREATE TABLE syntax

Basic syntax

```
CREATE TABLE <table name> (
        <attribute name 1> <data type 1>,
        ...
        <attribute name n> <data type n>);
```

- For the full syntax:

https://docs.microsoft.com/en-us/sql/t-sql/statements/create-table-transact-sql?view=sql-server-ver15

Create Table example

```
CREATE TABLE subject (
        Subject_id  INT IDENTITY(1,1) NOT NULL,
        Menu_item VARCHAR(255) NOT NULL,
        Position TINYINT,
        Visible BIT DEFAULT 1,
                PRIMARY KEY (Subject_id)
        );
```

IDENTITY Article
https://dotnettutorials.net/lesson/identity-column-sql-server/

ALTER TABLE

```
ALTER TABLE subject
        ADD cost FLOAT(2);
```

ALTER TABLE

```
ALTER TABLE <table name>
ADD CONSTRAINT <constraint name> PRIMARY KEY
(<attribute list>);
```

ALTER TABLE

- Statement may be used to specify **primary & foreign key constraints**, as well as to make other modifications to the table structure
- Key constraints may also be specified in the CREATE TABLE statement – but need to include a constraint name.
- You should specify the constraint name (e.g. con_customer_id). The attribute list contains one or more attributes
- if more than one, the names are separated by commas

PRIMARY KEY Examples

```
ALTER TABLE person
    ADD PRIMARY KEY (Customer_id);
```

```
ALTER TABLE person
    ADD pk_person PRIMARY KEY (Customer_id);
```

- For defining a **PRIMARY KEY constraint on multiple columns**:

```
ALTER TABLE person
    ADD CONSTRAINT pk_person PRIMARY KEY
        (Customer_id, Last_name);
```

UNIVERSITY OF PLYMOUTH

Foreign key

- Need to specify both the foreign key attributes in the (child) table & the primary key attributes they link to in the parent table

```
ALTER TABLE <table name>
        ADD CONSTRAINT <constraint name> FOREIGN
        KEY  (<attribute list>)
                REFERENCES <parent table name>
                        (<attribute list>);
```

Foreign key

- If there is more than one attribute in the FK, all of them must be included (with commas between) in both the FK attribute list **&** the REFERENCES (parent table) attribute list
- You need a separate foreign key definition for each relationship in which this table is the child

FOREIGN KEY Examples

```
ALTER TABLE order
        ADD FOREIGN KEY (Person_id)
        REFERENCES person (Person_id);
```

- For defining a FOREIGN KEY constraint on multiple columns:

```
ALTER TABLE order
        ADD CONSTRAINT FK_PersonOrder
        FOREIGN KEY (Person_id, Lastname)
        REFERENCES person (Person_id, Lastname);
```

UNIVERSITY OF PLYMOUTH

DROP

- You can delete any object you have created:

```
DROP TABLE <table name>;

ALTER TABLE <table_name>
    DROP COLUMN <column_name>;

ALTER TABLE <tablename>
    DROP CONSTRAINT <constraintname>;
```

DROP

- The **DROP TABLE** statement won’t work unless you separately drop any foreign keys that refer to the table you want to drop

- **It also removes all data that was contained in the table**
- Seems you can only drop the PRIMARY KEY by using the constraint name. Otherwise you would need to drop whole table & recreate it.

Data Manipulation Language (DML)

- DML is the subset of SQL used to add, retrieve, update & delete data

Operation	SQL	Description
Create	**INSERT INTO**	Inserts new data
Read (Retrieve)	**SELECT**	Extracts data
Update	**UPDATE**	Updates data
Delete	**DELETE**	Deletes data

INSERT INTO

- Adds new rows to a table:

```
INSERT INTO <table name>
     VALUES (<value1>, ..., <value n>);
```

The comma delimited list of values must match the table structure **exactly** in the number of attributes & the data type of each attribute – except for sequences, which are omitted.

You need a separate INSERT statement for every row

INSERT INTO

- Character type values are always enclosed in single quotes
- Numeric values are never in quotes
- Date values are often (but not always) in the format

  'yyyy-mm-dd‘

  e.g. '2006-11-30‘

However, there are functions which will perform a conversion to a date/time datatype

UPDATE

- Change values that are already in a table:

```
UPDATE <tablename>
    SET <attribute> = <expression> WHERE<condition>;
```

- The expression can be:

  a constant,

  any computed value,

  or even the result of a SELECT statement

  that returns a single row & a single column

UPDATE

- If the WHERE clause is omitted, then the specified attribute is set to the same value in **every row** of the table
- You can also set multiple attribute values at the same time with a comma-delimited list of attribute=expression pairs

DELETE FROM

- Deletes records (rows) in a table:

```
DELETE FROM <table name> WHERE <condition>;
```

- If the WHERE clause is omitted, then every row of the table is deleted

  - you will not get a “do you really want to do this?” message!

SUMMARY

- Write SQL statements that will build up a set of tables within an RDBMS implementation
- Write SQL statements that will manipulate the existing structures to add, modify and remove columns and constraints.

University of Plymouth

Advancing knowledge, transforming lives

UNIVERSITY OF
PLYMOUTH