



CITS1402 Relational Database Management Systems

Week 2—Introduction to SQL

Objectives

Purpose and importance of SQL.

Writing SQL Commands

How CREATE tables

How to specify PKs and FKs

How to DROP tables

Reference: Database Systems by Thomas Connolly

Chapter 6/7 - Objectives

Purpose and importance of SQL.

Writing SQL Commands

How CREATE tables

How to specify PKs and FKs

How to DROP tables



Will be discussed briefly
More stuff to come..

Chapter 6/7 - Objectives

Purpose and importance of SQL.

Writing SQL Commands

How CREATE tables

How to specify PKs and FKs

How to DROP tables

Objectives of SQL

Ideally, database language should allow user to:

- create the database and relation structures;**
- perform insertion, modification, deletion of data from relations;**
- perform simple and complex queries.**

Must perform these tasks with minimal user effort and command structure/syntax must be easy to learn.

It must be portable.

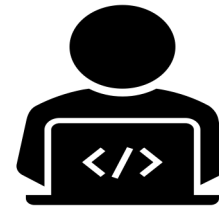
Objectives of SQL



SQL is a transform-oriented language with 2 major components:

A **DDL** for defining database structure.

A **DML** for retrieving and updating data.



Until SQL:1999, SQL did not contain flow of control commands. These had to be implemented using a programming or job-control language, or interactively by the decisions of user.

Objectives of SQL

SQL is relatively easy to learn:

- it is non-procedural -

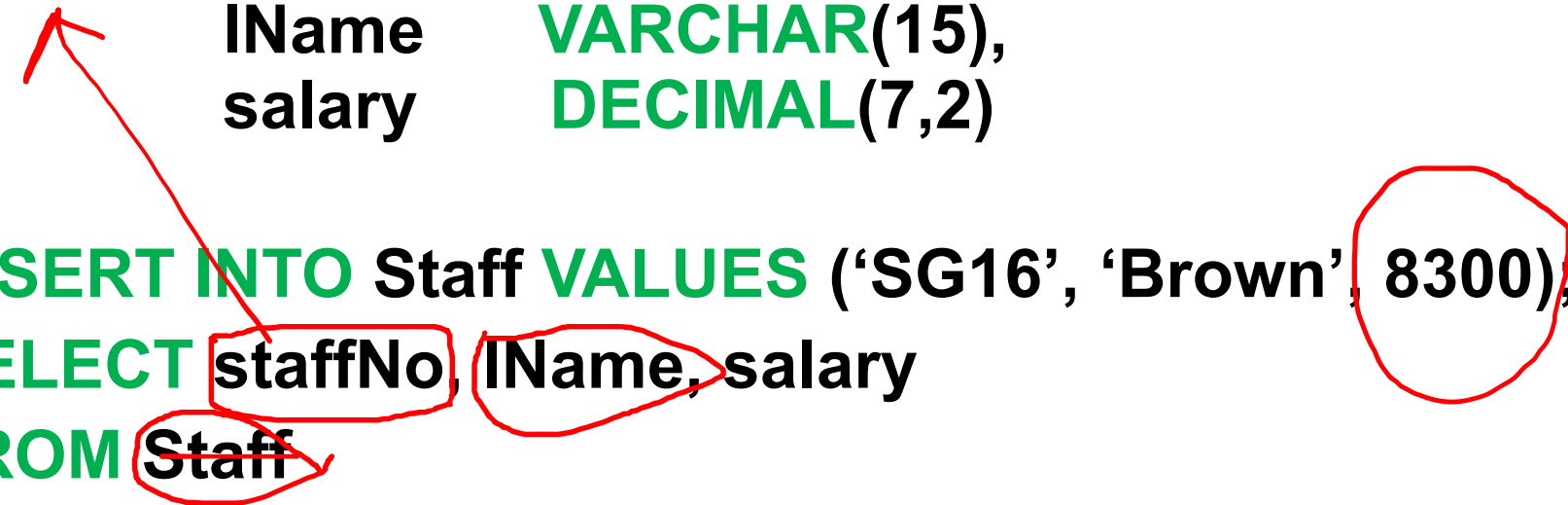
 - WHAT** information you require, rather than

 - HOW** to get it;

- it is essentially free-format.

Objectives of SQL

Consists of standard English words:

- 1) **CREATE TABLE** Staff(
 staffNo **CHAR**(5),
 IName **VARCHAR**(15),
 salary **DECIMAL**(7,2)
);
 - 2) **INSERT INTO** Staff **VALUES** ('SG16', 'Brown', 8300);
 - 3) **SELECT** **staffNo**, **IName**, salary
 FROM Staff
 WHERE salary > 10000;
- 

Objectives of SQL

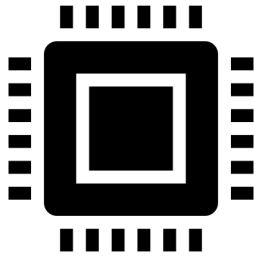


Can be used by range of users including DBAs, management, application developers, and other types of end users.



An ISO standard now exists for SQL, making it both the formal and *de facto* standard language for relational databases.

History of SQL



In 1974, D. Chamberlin (IBM San Jose Laboratory) defined language called

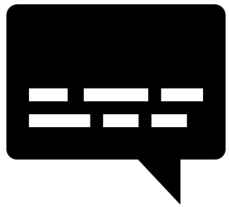


'Structured English Query Language'
(**SEQUEL**).

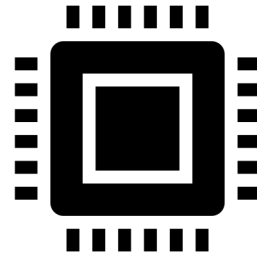


A revised version, **SEQUEL/2**, was defined in 1976 but name was subsequently changed to **SQL** for legal reasons.

History of SQL



Still pronounced 'see-quel', though official pronunciation is 'S-Q-L'.



IBM subsequently produced a prototype DBMS called *System R*, based on SEQUEL/2.



Roots of SQL, however, are in SQUARE (Specifying Queries as Relational Expressions), which predates System R project.

History of SQL

In late 70s, **ORACLE** appeared and was probably **first** commercial **RDBMS** based on SQL.

1987: ANSI and ISO published an initial standard for SQL.

1989: ISO published an addendum that defined an '**Integrity Enhancement Feature**'.

1992: first major revision to ISO standard occurred, referred to as SQL2 or SQL/92.

1999: SQL:1999 was released with support for **object-oriented data** management.

2003: SQL:2003, was released (**XML-related** features)

2006: SQL:2006, more XML-related features

2008, SQL:2008, new clauses

2011, SQL:2011, **temporal support**

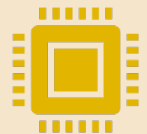
Importance of SQL



SQL has become part of application architectures such as IBM's Systems Application Architecture.



It is strategic choice of many large and influential organizations (e.g. X/OPEN).



SQL is Federal Information Processing Standard (FIPS) to which conformance is required for all sales of databases to American Government.

Importance of SQL

SQL is used in other standards and even influences development of other standards as a definitional tool. Examples include:

ISO's Information Resource Directory System (IRDS) Standard

Remote Data Access (RDA) Standard.

SQL has academic interest and adapting to new domains, e.g. OnLine Analytical Processing (OLAP)

Chapter 6/7 - Objectives

Purpose and importance of SQL.

Writing SQL Commands

How CREATE tables

How to specify PKs and FKs

How to DROP tables

Writing SQL Commands

SQL statement consists of *reserved words* and *user-defined words*.

- **Reserved words** are a fixed part of SQL and must be spelt exactly as required and cannot be split across lines.
- **User-defined** words are made up by user and represent names of various database objects such as relations, columns, views.

Writing SQL Commands

Most components of an SQL statement are *case insensitive*, except for literal character data.

Freeform, but...

More readable with indentation and lineation:

- Each clause should begin on a new line.

- Start of a clause should line up with start of other clauses.

- If clause has several parts, should each appear on a separate line and be indented under start of clause.

Writing SQL Commands

Example of how to indent SQL clauses

No indenting:

```
SELECT staffNo, IName, salary FROM Staff WHERE salary > 10000;
```

Good indenting:

```
SELECT staffNo, IName, salary  
FROM Staff  
WHERE salary > 10000;
```

Writing SQL Commands

Example of how to indent SQL clauses

No indenting:

```
SELECT staffNo, IName, salary FROM Staff WHERE salary > 10000;
```

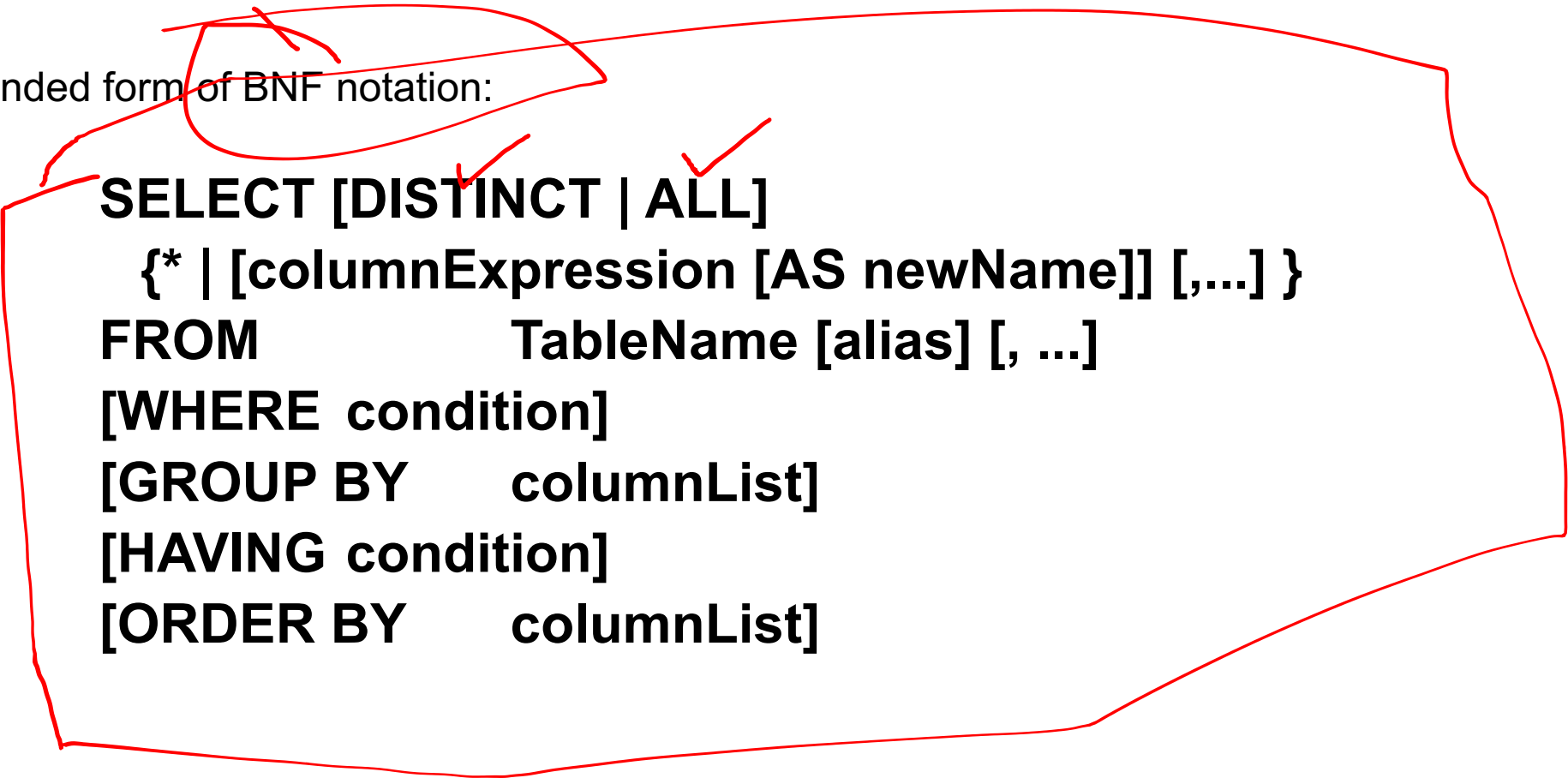
Good indenting:

```
SELECT staffNo, IName, salary  
FROM Staff  
WHERE salary > 10000 and  
      branchNo = 'B005';
```



Writing SQL Commands

Use extended form of BNF notation:



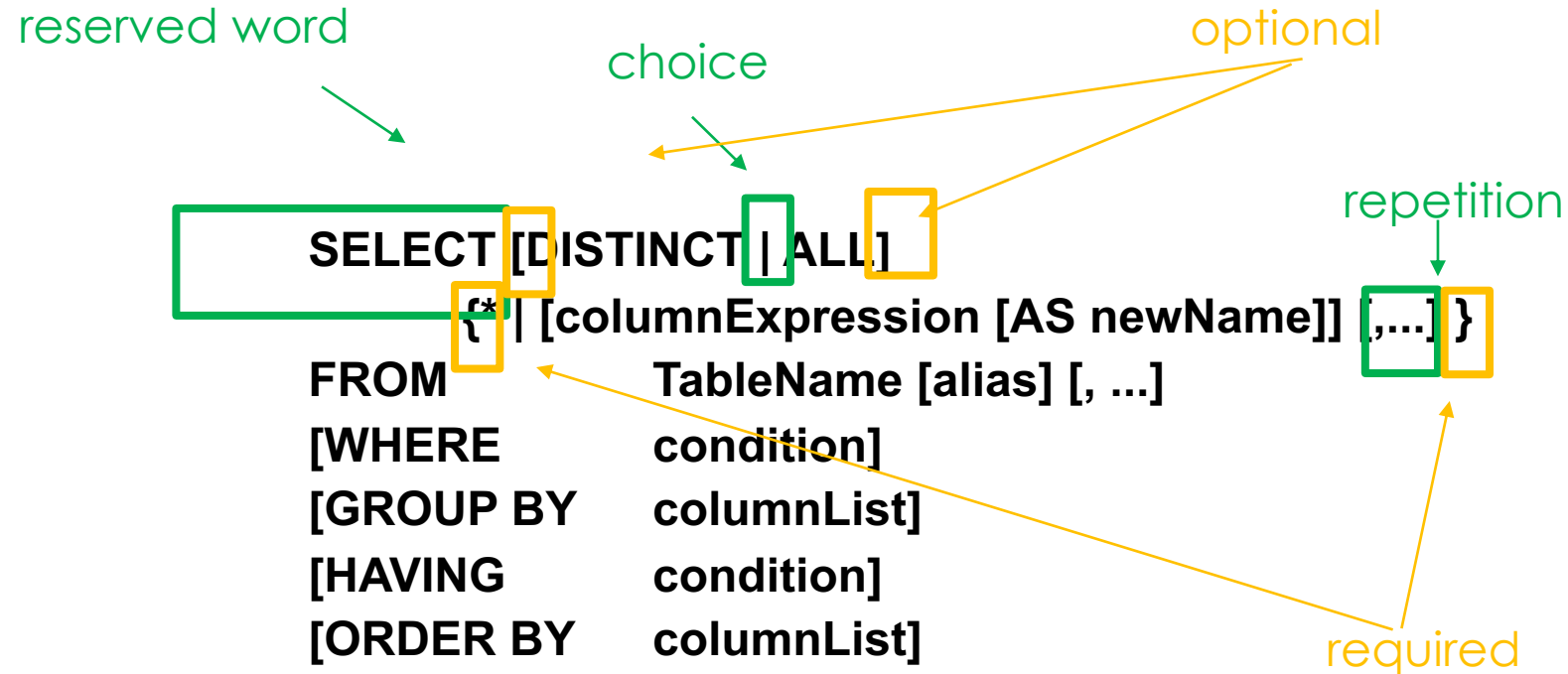
```
SELECT [DISTINCT | ALL]  
    { * | [columnExpression [AS newName]] [, ...] }  
FROM      TableName [alias] [, ...]  
[WHERE condition]  
[GROUP BY columnList]  
[HAVING condition]  
[ORDER BY columnList]
```

Writing SQL Commands

Use extended form of BNF notation:

- **Upper-case** letters represent **reserved** words.
- **Lower-case** letters represent **user-defined** words.
- **|** indicates a *choice* among alternatives.
- **Curly braces** indicate a *required element*.
- **Square brackets** indicate an *optional element*.
- **...** indicates *optional repetition* (0 or more).

SELECT Statement



Literals

Literals are constants used in SQL statements.

All **non-numeric literals** must be enclosed in single quotes

e.g. **'London'**

All **numeric literals** must not be enclosed in quotes

e.g. **650.00**

Chapter 6/7 - Objectives

Purpose and importance of SQL.

Writing SQL Commands

How CREATE tables

How to specify PKs and FKs

How to DROP tables

CREATE TABLE

```
CREATE TABLE TableName(  
  {colName dataType [NOT NULL] [UNIQUE]  
  [DEFAULT defaultOption] [CHECK searchCondition] [...]}  
→ [PRIMARY KEY (listOfColumns),]  
→ {[UNIQUE (listOfColumns)] [...]}  
→ {[FOREIGN KEY (listOfFKColumns)  
  → REFERENCES ParentTableName [(listOfCKColumns)],  
  → [MATCH {PARTIAL | FULL}]  
  [ON UPDATE referentialAction]  
  [ON DELETE referentialAction ]  
  [...]}  
→ {CHECK (searchCondition) [...] }
```

CREATE TABLE

```
CREATE TABLE TableName (  
    {colName dataType [NOT NULL] [UNIQUE]  
→ [DEFAULT defaultOption] [CHECK searchCondition] [...]}  
    [PRIMARY KEY (listOfColumns),  
→ {[UNIQUE (listOfColumns)] [...]}  
    {[FOREIGN KEY (listOfFKColumns)  
        REFERENCES ParentTableName [(listOfCKColumns)],  
        [MATCH {PARTIAL | FULL}]  
        [ON UPDATE referentialAction]  
        [ON DELETE referentialAction ]  
    } [...]}  
    {[CHECK (searchCondition)] [...] })
```

CREATE TABLE

Creates a table with one or more columns of the specified *dataType*.

With NOT NULL, system rejects any attempt to insert a null in the column.

Can specify a DEFAULT value for the column.

Primary keys should always be specified as NOT NULL.

FOREIGN KEY clause specifies FK along with the referential action.

CREATE TABLE - Staff

```
CREATE TABLE Staff (  
    staffNo      CHAR(4) NOT NULL,  
    fName        VARCHAR(50),  
    lName        VARCHAR(50),  
    position     VARCHAR(50),  
    sex          CHAR(1),  
    DOB          DATE,  
    salary       INT,  
    branchNo     CHAR(4) );
```

Primary Key – Entity Integrity

Entity Integrity - Primary key of a table must contain a unique, non-null value for each row.

ISO standard supports PRIMARY KEY clause in CREATE and ALTER TABLE statements:

PRIMARY KEY(staffNo)

PRIMARY KEY(clientNo, propertyNo)

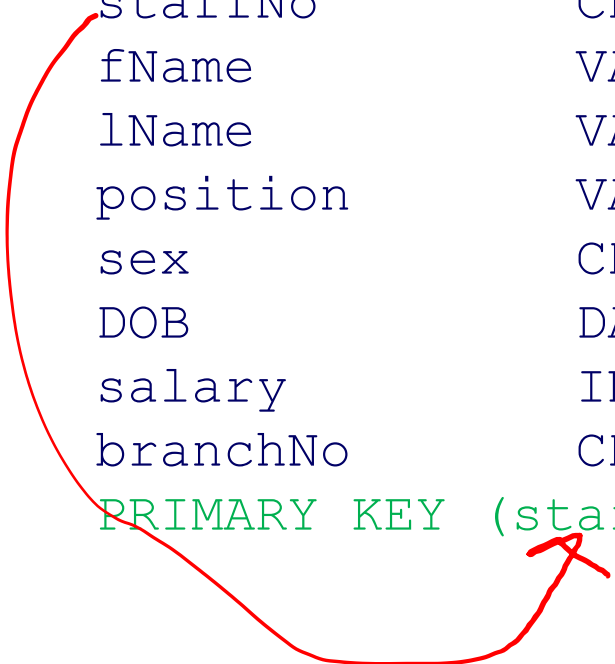
Can only have one PRIMARY KEY clause per table.

Can still ensure uniqueness for alternate keys using UNIQUE:

 **UNIQUE(telNo)**

CREATE TABLE - Staff

```
CREATE TABLE Staff (  
    staffNo      CHAR(4) NOT NULL,  
    fName        VARCHAR(50),  
    lName        VARCHAR(50),  
    position     VARCHAR(50),  
    sex          CHAR(1),  
    DOB          DATE,  
    salary       INT,  
    branchNo     CHAR(4),  
    PRIMARY KEY (staffNo));
```



Foreign Key (FK) - Referential Integrity

FK is column or set of columns that is specified and links each row in **child** table containing foreign FK to row of **parent** table containing matching PK.

Referential integrity means that, if FK contains a value, that value **must** refer to existing row in parent table.

ISO standard supports definition of FKs with FOREIGN KEY clause in CREATE and ALTER TABLE:

FOREIGN KEY(branchNo) REFERENCES Branch



CREATE TABLE - Staff

```
CREATE TABLE Staff (  
    staffNo      CHAR(4) NOT NULL,  
    fName        VARCHAR(50),  
    lName        VARCHAR(50),  
    position     VARCHAR(50),  
    sex          CHAR(1),  
    DOB          DATE,  
    salary       INT,  
    branchNo     CHAR(4),  
    PRIMARY KEY (staffNo),  
    FOREIGN KEY (branchNo) REFERENCES Branch (branchNo));
```


Foreign Key - Referential Integrity

Staff **Pk**

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

PrivateOwner

ownerNo	fName	lName	address	telNo	eMail	password
CO46	Joe	Keogh	2 Fergus Dr, Aberdeen AB2 7SX	01224-861212	jkeogh@lhh.com	*****
CO87	Carol	Farrel	6 Achray St, Glasgow G32 9DX	0141-357-7419	cfarrel@gmail.com	*****
CO40	Tina	Murphy	63 Well St, Glasgow G42	0141-943-1728	tinam@hotmail.com	*****
CO93	Tony	Shaw	12 Park Pl, Glasgow G4 0QR	0141-225-7025	tony.shaw@ark.com	*****

PropertyForRent

propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93	SG14	B003

Foreign Key - Referential Integrity

Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SI 21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

What happens if Ann Beech is fired? Delete from SG37

What happens if Joe Keogh leaves?

PrivateOwner

ownerNo	fName	lName	address	telNo	eMail	password
CO46	Joe	Keogh	2 Fergus Dr, Aberdeen AB2 7SX	01224-861212	jkeogh@lhh.com	*****
CO87	Carol	Farrel	6 Achray St, Glasgow G32 9DX	0141-357-7419	cfarrel@gmail.com	*****
CO40	Tina	Murphy	63 Well St, Glasgow G42	0141-943-1728	tinam@hotmail.com	*****
CO93	Tony	Shaw	12 Park Pl, Glasgow G4 0QR	0141-225-7025	tony.shaw@ark.com	*****

PropertyForRent

propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93	SG14	B003

Foreign Key - Referential Integrity

Any INSERT/UPDATE attempting to create FK value in child table **without** matching CK value in parent is **rejected**.

Action taken attempting to **update/delete** a CK value in **parent table** with matching rows in child is dependent on referential action specified using **ON UPDATE** and **ON DELETE** subclauses:

CASCADE	- SET NULL
SET DEFAULT	- NO ACTION

Foreign Key - Referential Integrity

FOREIGN KEY (staffNo) REFERENCES Staff ON DELETE SET NULL

FOREIGN KEY (ownerNo) REFERENCES Owner ON UPDATE CASCADE

Foreign Key - Referential Integrity

CASCADE: Delete row from parent and **delete matching rows in child**, and so on in cascading manner.

SET NULL: Delete row from parent and **set FK column(s) in child to NULL**. Only valid if FK columns are NOT NULL.

SET DEFAULT: Delete row from parent and set each component of **FK in child to specified default**. Only valid if DEFAULT specified for FK columns.

NO ACTION: **Reject delete** from parent. **DEFAULT**.

CREATE TABLE - Staff

```
CREATE TABLE Staff (  
    staffNo      CHAR(4) NOT NULL,  
    fName        VARCHAR(50),  
    lName        VARCHAR(50),  
    position     VARCHAR(50),  
    sex          CHAR(1),  
    DOB          DATE,  
    salary       INT,  
    branchNo     CHAR(4),  
    PRIMARY KEY (staffNo),  
    FOREIGN KEY (branchNo) REFERENCES Branch (branchNo)  
        ON UPDATE CASCADE ON DELETE SET NULL);
```

Objectives

Purpose and importance of SQL.

Writing SQL Commands

How CREATE tables

How to specify PKs and FKs

How to DROP tables

DROP TABLE

DROP TABLE TableName [RESTRICT | CASCADE]

e.g. DROP TABLE PropertyForRent;

Removes named table and all rows within it.

RESTRICT, if any other objects depend for their existence on continued existence of this table, SQL **does not allow request**.

Order of table drops can be dependent on FKs

CASCADE, SQL **drops all dependent objects** (and objects dependent on these objects).