Database Basics/Relational Model/Conceptual Model/Logical Model

- Data redundancy: When the same data is stored unnecessarily at different places
- Data anomaly: Problems that can occur in poorly planned databases
    - Insertion anomaly: Not possible to add data unless another piece of data is also added
    - Deletion anomaly: Deleting a record results in deletion of other required data
    - Update anomaly: May have to change many records (some changes may be made incorrectly)
- Need to remember the 8 relational set operators (select, project, union, intersect, difference, product)
- Selection has the sigma sign, projection has pi
- Functional dependence: The value of one or more attributes determines the value of one or more attributes
- Full functional dependence: When entire collection of attributes in determinant is necessary for the relationship
- Primary key: Candidate key that can uniquely identify all other attribute values in a given row
- Secondary key: Key strictly for data retrieval purposes (not unique)
- Super key: Attribute or group of attributes that can uniquely identify any row in the table
- Candidate key: A minimal super key
- ANSI/SPARC is comprised of 3 levels external (requirements definition), conceptual (conceptual design and logical design), internal level (physical design)
- Conceptual design only has keys and can have relationships (havent decided which database model to use RMDBS, NoSQL)
- Logical design identifies PK, FK, resolves all M:N relationships (have decided which DB model to use (relational, object-oriented BUT have not decided which DBMS vendor)
- Physical design: Physical implementation of the logical data model (dependent on the DBMS vendor like Oracle, MySQL)
- A derived attribute does not need to be stored in the database
- Existence-independent attribute: When an attribute can exist apart from other entities
- Mapping 1:1 binary relationship: PK on mandatory side becomes FK on optional side of relationship
- Mapping 1:1 binary: If both are optional, choose the side that causes the least nulls. If both are mandatory, consider combining into single entity
- Bridge entity know as composite entity
- JOIN is NOT a set operator
- For a relational model, the order of the attributes doesn't matter
- ANSI/SPARC levels are external level (requirements definition), conceptual level (conceptual design and logical design), internal level (physical design)
- Logical model: Need to include action verbs and data types for each column
- Logical model: Can put surrogate keys for anything
- Normalisation: Can shorten terms from like organisation_contact to o_contact
- Normalisation: o_contact would be the PK for the o_contact, o_name -> FOLLOW BUSINESS RULES BETTER
- Logical model: Split up into as many entities as appropriate

Relational Algebra

- Natural join: cartesian product of two tables, select to find matching tuples, project away duplicate columns
- Equijoin is basically a join on
- Relational algebra: Try to delay your JOIN operations for efficiency
- Relational algebra: Can be super efficient by projecting to take only the values that matter
- Relational algebra: Distinct rows are automatically removed after projection

- Theta-joins are for combining rows as long as they fit a condition
- If a thetajoin uses a = comparator, it is an equijoin
- Natural join uses no comparison operator, removes that duplicate column
- Inner join: Rows with matching attributes are included, rest is discard
- Outer -join: All tuples from relations are included

Normalisation

- 1NF: No repeating groups, PK identiifed
- 2NF no partial dependencies
- 3NF no transitive dependencies
- When designing database -> Need to make sure entities are already in normal form
- Need to include primary key in relations when normalisation

SQL

- When using predicates in WHERE to evaluate values -> NULL values are not returned
- Table Aliasing: FROM unit u
- Column aliasing: SELECT marks as lol
- Number of rows after cartesian product is rows(first table) X rows(second table)
- Can do cartesian product by select * from table1, table2
- Executing a DDL statement after a DML will commit the DML statement too
- Cannot have group functions in WHERE, need to put it in HAVING
- When you include a group function, need to have all non-group functions in GROUP BY clause
- Can use COUNT(DISTINCT(_)) to count the total number of distinct entries
- Take time to visualise the initial tables before shrinking it with conditions
- MAX returns one value, usually need to include it as a subquery
- When comparing two dates, the older date is smaller than the newer date
- e.g. Questions like finding avg or max of something of a group of entries -> Tend to use the same conditions throughout both
- Attributes used in SELECT, HAVING and ORDER BY needs to be included in GROUP BY
- Can use correlated subqueries by putting an attribute from the outer query inside in the inner query
- Can create views using CREATE VIEWS _ AS SELECT __
- Can update views (not using group functions or set operators) only if PK of base table is still preserved in view
- Union-compatible is when the names of the attributes can be different but the data types must be the same
- NVL(result, string) converts any null values to a given string
- Can do TO_CHAR(number, '0.00') to compare number formats
- In UPDATE OF [columns], can include multiple columns
- Can use IF UPDATING THEN, or ELSE IF or ELSE as control structure
- WHEN clause can only be used by row-level triggers, don't need to include : for old and new
- Can do string comparisons using LIKE and wildcards % and _
- Should do ORDER BY for everything
- Can do a JOIN ON with value A between B and C (more than just equal)
- Tend to do a LEFT OUTER JOIN or RIGHT OUTER JOIN when you want to show all entries of something as compared to the other table
- initcap(string) used to bring stuff to Initial Capitalisation
- If you wanna count a general occurence you can do COUNT(*)
- HAVING allows you to use aggregate functions on LHS
- Distinct applies to the whole row as being unique
- Can get day by to_char(date, 'day') and month by 'month', year by 'year'

- When specifying problems with an associated script, need to mention the error message/location
- Don't need to check if something is NULL in CHECK
- Auditing application: auditors can record when an employee leaves by placing employee entries into a EMPLOYEE table
- WHERE clause applied to all rows, HAVING applied to groups defined by GROUP BY
- Can do CHECK to make constraints
- Can use ALTER TABLE [MODIFY, DROP, ADD][CONSTRAINT]
- Main difference between count(*) and count(attr) is that count(*) counts nulls too
- ALL = AND operator for every value
- ANY = OR operator for every value
- For a view, the data in a view is generated only when the view is used

Transaction Management

- Transaction is a logical unit of work that must be either all completed or all aborted
- Consistent database state is one where all the data integrity constraints are satisfied
- Wait-For graph waits for the first transaction still using something
- Soft-crashes: loss of volatile storage but no damage to disks -> need to restart
- Hard crash: disk is unreadable -> Needs recovery
- ACID questions: Need to go through multiple properties
- Lost updates are when two transactions executed concurrently are updating the same data and one of the updates is overwritten
- Uncommitted data: When one transaction reads some data after a transaction updates but rolls back, first transaction is still using the uncommitted data
- Inconsistent retrieval: when a transaction uses summary functions over some data while another transaction is updating the data
-

Database Connectivity and Future Technologies

- Database middleware: manages connectivity and data transformation issues (eg. MS ODBC, JDBC)
- Web-to-database middleware: Program that interact directly with the server process to handle specific types of requests
    - Web server <> Web-to-database-middleware <> Database middleware <> Database
- NoSQL: Property 1- distributed (large clusters), property 2 -no update, append only so it is optimised for a main operation
- PHP is a server- side language that can connect to the Oracle database using the OCI8 extension
    - oci_connect used to connect to an Oracle Database
- Web frameworks: Provides easier development of web apps
    - Also support Oracle connectivity
- Object-Relational Mapping: Allows queries and data manipulation with a database using OOP
- SQL Injection: Inserting an SQL query via input from the client to the app for harmful use
    - Prevented by sanitising and checking the input