Data Analytics

Level 1: Writes SQL code to generate desired effects on a relational database

- [x] Anticipates the result of SQL queries executed against relation instances
- [x] Expresses declarative query intent in terms of relational operators (Find in NoasQueries.sql)
- [x] Extracts data from relations with precise selection predicates and attribute projection
- [x] Combines data from tables with appropriately chosen JOIN operations
- [x] Modifies data in relations in bulk with set-theoretic SQL DML queries

Level 2: Massages data into visualisation-ready layouts by slicing, dicing, pivoting, and rolling it up it directly in SQL

- [x] Expresses complex logic as single SQL queries using aggregation and sub-queries
- [x] Understands how functional dependencies and referential integrity affect the semantics of queries
- [ ] Describes the logical ordering of operators in complex queries that involve nested logic
- [ ] Plans out how to transform data from relations into a desired output layout as in standard OLAP/ETL operators
- [ ] Prefers embedding complex logic into RDBMS over handling it in application-layer code

Level 3:Uses a variety of SQL constructs and indexes to produce readable, efficient, idiomatic queries

- [x] Avoids sub-queries when possible.
- [x] Identifies which attributes should be indexed in order to accelerate a query.
- [x] Embraces declarative aspects of SQL to write concise code and avoids iteration logic whenever possible.
- [x] Styles code in a manner consistent with the broader SQL developer community
- [x] Articulates what makes one query better than another semantically equivalent query

Level 4:Optimises SQL queries to map onto more efficient physical operators

- [x] Creates indexes that prevent the materialisation of temporary tables.
- [x] Recognises queries that will have poor asymptotic complexity in the external memory model
- [ ] Understands which logical operators can be rearranged in a query execution plan
- [ ] Appreciates why SQL is only partly declarative in practice and how this influences query design
- [x] Connects the physical layout of tables and indexes to the implementation of physical query operators

Data Modeling

Level 1: Stores data in a set of tables that are compatible with data sources

- [x] Selects appropriate data types for tables
- [x] Writes SQL code that implements a relational design
- [x] Loads data from .JSON and .CSV formats without truncation or other forms of data loss
- [x] Appends newly acquired data to pre-existing tables, modifying their structure as necessary
- [x] Documents the relationships between tables with syntactically and semantically correct entity-relationship diagrams

Level 2: Constructs well-normalised conceptual and relational schemata that capture requirements without redundancy

- [x] Eliminates data anomalies with effective normalisation
- [x] Identifies dependencies among attributes and appropriate identifiers/keys for entity sets and relations
- [x] Justifies the quality of a schema through a theoretical lens (ERD, Normalization, Algebra)
- [x] Maps requirements onto schemata and vice versa to ensure designs are minimal and complete
- [x] Internalises the merits of the relational data model even still today

Level 3: Applies advanced ERD constructs and normalisation methods to produce more natural schemata

- [x] Uses inheritance and weak entity sets when they are more expressive than alternatives.
- [x] Assesses incongruity between conceptual and relational schemata.
- [ ] Applies alternative normal forms when they better suit the application requirements.
- [ ] Simplifies complicated relationships with powerful ERD constructs like ternary relationships, identifiers on relationships, and composite attributes
- [x] Systematically evaluates strengths and weaknesses of a schema using a consistent framework
- [x] Considers the impact of NULL values and inheritance on functional dependencies

Level 4:Designs industrial-strength databases that can be deployed in the real world

- [ ] Designs for extensibility with an appreciation that data sources may change
- [x] Considers matters of data governance, ethics, and data privacy in deciding how to meet application requirements
- [ ] Anticipates data access patterns and load in database design and carefully considers trade-offs like denormalisation
- [ ] Develops conceptual schemata that are compatible with multiple logical database models
- [x] Avoids over-engineering designs in favour of simplicity and satisfying only those requirements already known