Florida Department of Education
Curriculum Framework

Program Title:	**Database Application Development & Programming**
Program Type:	**Career Preparatory**
Career Cluster:	**Information Technology**

Career Certificate Program	
Program Number	Y700300
CIP Number	0511020315
Grade Level	30, 31
Standard Length	1200 hours
Teacher Certification	Refer to the **Program Structure** section.
CTSO	Phi Beta Lambda BPA SkillsUSA
SOC Codes (all applicable)	15-1151 – Computer User Support Specialists 15-1131 – Computer Programmers
CTE Program Resources	http://www.fldoe.org/academics/career-adult-edu/career-tech-edu/program-resources.stml
Basic Skills Level	Mathematics: 9 Language: 9 Reading: 9

Purpose

This program offers a sequence of courses that provides coherent and rigorous content aligned with challenging academic standards and relevant technical knowledge and skills needed to prepare for further education and careers in the Information Technology career cluster; provides technical skill proficiency, and includes competency-based applied learning that contributes to the academic knowledge, higher-order reasoning and problem-solving skills, work attitudes, general employability skills, technical skills, and occupation-specific skills, and knowledge of all aspects of the Information Technology career cluster.

The content includes but is not limited to the fundamentals of programming and software development; procedural and object-oriented programming; creating regular and specialized applications using standard and extended Structured Query Language (SQL), including testing, monitoring, debugging, documenting, and maintaining database applications.

Additional Information relevant to this Career and Technical Education (CTE) program is provided at the end of this document.

Program Structure

This program is a planned sequence of instruction consisting of four occupational completion points.

This program is comprised of courses which have been assigned course numbers in the SCNS (Statewide Course Numbering System) in accordance with Section 1007.24 (1), F.S. Career and Technical credit shall be awarded to the student on a transcript in accordance with Section 1001.44 (3)(b), F.S.

To teach the courses listed below, instructors must hold at least one of the teacher certifications indicated for that course.

The following table illustrates the postsecondary program structure:

OCP	Course Number	Course Title	Teacher Certification	Length	SOC Code
A	OTA0040	Information Technology Assistant	OTA0040 Teacher Certifications	150 hours	15-1151
B	CTS0041	Computer Programmer Assistant	BUS ED 1 @2 COMPU SCI 6 COMP PROG 7G	300 hours	15-1131
C	CTS0044	Computer Programmer		150 hours	15-1131
D	CTS0062	Database Programmer		600 hours	15-1131

<u>Common Career Technical Core – Career Ready Practices</u>

Career Ready Practices describe the career-ready skills that educators should seek to develop in their students. These practices are not exclusive to a Career Pathway, program of study, discipline or level of education. Career Ready Practices should be taught and reinforced in all career exploration and preparation programs with increasingly higher levels of complexity and expectation as a student advances through a program of study.

1. Act as a responsible and contributing citizen and employee.

2. Apply appropriate academic and technical skills.

3. Attend to personal health and financial well-being.

4. Communicate clearly, effectively and with reason.

5. Consider the environmental, social and economic impacts of decisions.

6. Demonstrate creativity and innovation.

7. Employ valid and reliable research strategies.

8. Utilize critical thinking to make sense of problems and persevere in solving them.

9. Model integrity, ethical leadership and effective management.

10. Plan education and career path aligned to personal goals.

11. Use technology to enhance productivity.

12. Work productively in teams while using cultural/global competence.

Standards

Information Technology Assistant (OTA0040) is the first course in this and other programs within the Information Technology Career Cluster. Standards 01.0 – 14.0 are associated with this course.

After successfully completing this program, the student will be able to perform the following:

01.0 Demonstrate knowledge, skill, and application of information systems to accomplish job objectives and enhance workplace performance.
02.0 Develop an awareness of microprocessors and digital computers.
03.0 Demonstrate an understanding of operating systems.
04.0 Use technology to enhance the effectiveness of communication skills utilizing word processing applications.
05.0 Use technology to enhance communication skills utilizing presentation applications.
06.0 Use technology to enhance the effectiveness of communication utilizing spreadsheet and database applications.
07.0 Use technology to enhance communication skills utilizing electronic mail.
08.0 Investigate individual assessment and job/career exploration and individual career planning that reflect the transition from school to work, lifelong learning, and personal and professional goals.
09.0 Incorporate appropriate leadership and supervision techniques, customer service strategies, and standards of personal ethics to accomplish job objectives and enhance workplace performance.
10.0 Demonstrate competence using computer networks, internet and online databases to facilitate collaborative or individual learning and communication.
11.0 Demonstrate competence in page design applicable to the WWW.
12.0 Develop an awareness of emerging technologies.
13.0 Develop awareness of computer languages and software applications.
14.0 Demonstrate comprehension and communication skills.
15.0 Use oral and written communication skills in creating, expressing and interpreting information and ideas.
16.0 Explore the characteristics, tasks, work attributes, options, and tools associated with a career in software development.
17.0 Demonstrate an understanding of the characteristics, use, and selection of numerical, non-numerical, and logical data types.
18.0 Distinguish between iterative and non-iterative program control structures.
19.0 Differentiate among high level, low level, procedural, object-oriented, compiled, interpreted, and translated programming languages.
20.0 Describe the processes, methods, and conventions for software development and maintenance.
21.0 Explain the types, uses, and limitations of testing for ensuring quality control.
22.0 Create a program design document using Unified Modeling Language (UML) or other common design tool.
23.0 Solve problems using critical thinking skills, creativity and innovation.
24.0 Use information technology tools.
25.0 Describe the importance of security and privacy information sharing, ownership, licensure and copyright.
26.0 Design a computer program to meet specific physical, operational, and interaction criteria.
27.0 Create and document a computer program that uses a variety of internal and control structures for manipulating varied data types.
28.0 Create and document an interactive computer program that employs functions, subroutines, or methods to receive, validate, and process user input.
29.0 Effectively communicate and collaborate.
30.0 Demonstrate responsible use of technology and information.
31.0 Explain key concepts that distinguish object-oriented programming from procedural programming.

32.0 Create a project plan that defines requirements, structural design, time estimates, and testing elements.
33.0 Design, document, and create object-oriented computer programs.
34.0 Design a unit test plan for an object-oriented computer program, test and debug the program, and report the results.
35.0 Understand human interactions in intelligence.
36.0 Develop an awareness of the changes taking place in the information age and how they fit into an evolving society.
37.0 Develop the "big picture" of database design and how to best organize data according to business rules and/or client needs.
38.0 Develop the process of creating an entity by identifying relationships.
39.0 Formulate and assemble initial entity relationship by expanding on modeling concepts.
40.0 Consider the degree and optionality of relationships of entities.
41.0 Demonstrate proficiency in early construction stages of the data modeling process by using unique identifiers and many-to-many (M:M) relationships for building entity relationship diagrams.
42.0 Demonstrate proficiency in advanced data constructs by analyzing business requirements and diagramming entities and relationships.
43.0 Demonstrate proficiency in designing and adding complexity to an entity-relationship model (ERM).
44.0 Apply complex ERM information by fine-tuning entities and the process for relating them.
45.0 Apply initial database design and normalization by following the set of house rules that determine how items are stored and retrieved.
46.0 Demonstrate proficiency in the technique of normalization by labeling and organizing all items in a database in such a way as to prevent any confusion or mistakes.
47.0 Demonstrate proficiency in table normalization by combining the techniques of an entity relationship model or a top-down, business approach to data with normalization or a bottom-up mathematical approach to data.
48.0 Apply blueprint principles to begin designing a tool for creating a web-based interface access to a database.
49.0 Extend the logical model presentation model by normalizing the data and mapping the management system.
50.0 Apply techniques for building a storage management system by creating a website using templates and wizards.
51.0 Demonstrate design and functionality by constructing a group business presentation.
52.0 Demonstrate comprehension of database modeling competency through group presentation.
53.0 Demonstrate comprehension that the database management software is a system for organizing the storage unit (or database) according to business needs and rules, through data integrity constraints.
54.0 Demonstrate comprehension of aspects of SQL language interface by writing basic SQL statements.
55.0 Demonstrate proficiency working with columns, characters, and rows in SQL.
56.0 Demonstrate proficiency in using SQL comparison operators.
57.0 Demonstrate proficiency in using logical comparisons and precedence rules.
58.0 Demonstrate proficiency using SQL single row functions.
59.0 Demonstrate proficiency displaying data from multiple tables.
60.0 Demonstrate proficiency aggregating data using group functions.
61.0 Demonstrate proficiency utilizing subqueries.
62.0 Demonstrate proficiency producing readable output with SQL language interface, reporting tool, and data manipulation language.
63.0 Demonstrate proficiency creating and managing database objects.
64.0 Demonstrate proficiency altering tables and constraints implementing views.
65.0 Demonstrate mastery of creating and implementing views, synonyms, indexes and other database objects.
66.0 Demonstrate ability to control user access and SQL language interface and reporting tool.
67.0 Demonstrate comprehension of bundling features of SQL.
68.0 Demonstrate comprehension working with composite data types by writing executable script files.
69.0 Describe the differences between SQL and SQL extension languages.

70.0 Create program blocks.
71.0 Use variables in program blocks.
72.0 Recognize lexical units.
73.0 Recognize data types.
74.0 Use scalar data types.
75.0 Use various types of joins.
76.0 Use SQL group functions and subqueries.
77.0 Write executable statements.
78.0 Use nested blocks and variable scope.
79.0 Use good programming practices.
80.0 Write DML statements to manipulate data.
81.0 Retrieve data.
82.0 Manipulate data.
83.0 Use transaction control statements
84.0 Use IF conditional control statements.
85.0 Use CASE conditional control statements.
86.0 Use basic LOOP iterative control statements.
87.0 Use WHILE and FOR loop iterative control statements.
88.0 Use nested loop iterative control statements.
89.0 Use explicit cursors.
90.0 Use explicit cursor attributes.
91.0 Use cursor FOR loops.
92.0 Use cursors with parameters.
93.0 Use cursors for update transactions.
94.0 Use multiple cursors.
95.0 Handle exceptions.
96.0 Trap server exceptions.
97.0 Trap user-defined exceptions.
98.0 Create procedures.
99.0 Use parameters in procedures.
100.0 Pass parameters.
101.0 Create stored functions.
102.0 Use functions in SQL statements.
103.0 Manage procedures and functions.
104.0 Manage object privileges.
105.0 Use invoker’s rights.
106.0 Create packages.
107.0 Manage package constructs.
108.0 Use advanced package concepts.
109.0 Manage persistent state of package variables.
110.0 Use vendor-supplied packages.
111.0 Understand dynamic SQL.

112.0 Understand triggers.
113.0 Create DML triggers.
114.0 Create DDL and database event triggers.
115.0 Manage triggers.
116.0 Use large object data types.
117.0 Manage binary types.
118.0 Manage indexes.
119.0 Manage dependencies.
120.0 Demonstrate an understanding of Agile Development.
121.0 Program a database application.
122.0 Utilize the basic concepts of database design.
123.0 Utilize SQL and union queries.
124.0 Implement program statements using objects.
125.0 Utilize debugging tools and write error handlers.
126.0 Demonstrate file I/O.
127.0 Create forms and identify all the properties of a form.
128.0 Manipulate data using object models.
129.0 Develop custom controls.
130.0 Utilize API functions.
131.0 Demonstrate and implement database replication using programming tools.
132.0 Analyze and implement security options.
133.0 Implement client/server applications.
134.0 Optimize the performance of a database.
135.0 Perform application distribution.
136.0 Test and debug databases.
137.0 Describe the difference between relational and NoSQL databases.
138.0 Demonstrate an understanding of Data Science and the concept of Data mining.

Florida Department of Education
Student Performance Standards

Program Title: Database Application Development & Programming
Career Certificate Program Number: Y700300

Course Number: OTA0040 **Occupational Completion Point: A** **Information Technology Assistant – 150 Hours – SOC Code 15-1151**
Information Technology Assistant (OTA0040) is part of several programs across the various CTE career clusters. To ensure consistency, the standards and benchmarks for this course (01.0 – 14.0) have been placed in a separate document. To access this document, visit: Information Technology Assistant (OTA0040) - (RTF)

Course Number: CTS0041 **Occupational Completion Point: B** **Computer Programmer Assistant – 300 Hours – SOC Code 15-1131**		
15.0	Use oral and written communication skills in creating, expressing and interpreting information and ideas. The student will be able to:	
	15.01	Select and employ appropriate communication concepts and strategies to enhance oral and written communication in the workplace.
	15.02	Locate, organize and reference written information from various sources.
	15.03	Construct writings and/or communications using developmentally appropriate terminology.
	15.04	Interpret verbal and nonverbal cues/behaviors that enhance communication.
	15.05	Analyze the positive and negative impacts of technology on popular culture and personal life.
	15.06	Discuss how technology has changed the way people build and manage organizations and how technology impacts personal life.
	15.07	Evaluate ways in which adaptive technologies may assist users with special needs.
	15.08	Explain how societal and economic factors are affected by access to critical information.
	15.09	Discuss the challenges (e.g., political, social, and economic) in providing equal access and distribution of technology in a global society.
16.0	Explore the characteristics, tasks, work attributes, options, and tools associated with a career in software development. The student will be able to:	
	16.01	Explore a variety of careers to which computing is central.
	16.02	Compare and contrast appropriate and inappropriate social networking behaviors.
	16.03	Discuss the impact of computing on business and commerce (e.g., automated inventory processing, financial transactions, e-commerce, virtualization, cloud computing).
	16.04	Evaluate the impacts of irresponsible use of information (e.g., plagiarism, falsification of data) on collaborative projects.
	16.05	Identify tasks performed by programmers.
	16.06	Describe how businesses use computer programming to solve business problems.
	16.07	Investigate job opportunities in the programming field.
	16.08	Explain different specializations and the related training in the computer programming field.
	16.09	Explain the need for continuing education and training of computer programmers.
	16.10	Understand and identify ways to use technology to support lifelong learning.
	16.11	Explain enterprise software systems and how they impact business.

	16.12 Describe ethical responsibilities of computer programmers.
	16.13 Describe the role of customer support to software program quality.
	16.14 Identify credentials and certifications that may improve employability for a computer programmer.
	16.15 Identify devices, tools, and other environments for which programmers may develop software.
17.0	Demonstrate an understanding of the characteristics, use, and selection of numerical, non-numerical, and logical data types. The student will be able to:
	17.01 Identify the characteristics (e.g., size, limits) and uses of different numerical and non-numerical data types.
	17.02 Explain the types and uses of variables in programs.
	17.03 Determine the best data type to use for given programming problems.
	17.04 Compare and contrast simple data structures and their uses.
	17.05 Identify the types of operations that can be performed on different data types.
	17.06 Evaluate arithmetic and logical expressions using appropriate operator precedence.
	17.07 Explain how computers store different data types in memory.
	17.08 Demonstrate the difference between "data" and "information".
	17.09 Use different number systems to represent data.
	17.10 Explain how national and international standards (i.e., ASCII, UNICODE) are used to represent non-numerical data.
	17.11 Use Boolean logic to perform logical operations.
18.0	Distinguish between iterative and non-iterative program control structures. The student will be able to:
	18.01 Create non-iterative programming structures and explain their uses.
	18.02 Create iterative programming structures and explain their uses.
	18.03 Explain how sequence, selection, and iteration are building blocks of algorithms.
19.0	Differentiate among procedural, object-oriented, compiled, interpreted, and translated programming languages. The student will be able to:
	19.01 Differentiate between multiple levels of an operating system, translation, and interpretation that support program execution.
	19.02 Explain the program execution process (by an interpreter and in CPU hardware).
	19.03 Describe object-oriented concepts.

	19.04	Explain the characteristics of procedural and object-oriented programming languages.
	19.05	Compare and contrast programming languages that are compiled, interpreted, and translated.
	19.06	Classify programming languages by paradigm and application domain (e.g., imperative, functional, logic languages and how well suited they are for certain application domains such as web programming, symbolic processing, data/numerical processing).
20.0	Describe the processes, methods, and conventions for software development and maintenance. The student will be able to:	
	20.01	Describe a software development process that is used to solve problems at different software development stages.
	20.02	Describe and demonstrate ethical and responsible use of modern communication media and devices.
	20.03	Define alternative methods of program development (e.g., rapid prototyping, waterfall, spiral model, peer coding).
	20.04	List and explain the steps in the program development cycle.
	20.05	Describe different types of documentation used in the program development cycle (*e.g.*, requirements document, program design documents, test plans).
	20.06	Describe different methods used to facilitate version control.
21.0	Explain the types, uses, and limitations of testing for ensuring quality control. The student will be able to:	
	21.01	Explain the uses and limits of testing in ensuring program quality.
	21.02	Explain testing performed at different stages of the program development cycle (*e.g.,* unit testing, system testing, user acceptance testing).
	21.03	Describe and identify types of programming errors.
	21.04	Analyze and manipulate data collected by a variety of data collection techniques.
	21.05	Explain what tools are applied to provide automated testing environments.
22.0	Create a program design document using common design tool. The student will be able to:	
	22.01	Describe different design methodologies and their uses (*e.g.*, object-oriented design, structured design, rapid application development).
	22.02	Describe tools for developing a program design (*e.g.*, Unified Modeling Language, flowcharts, design documents, pseudocode).
	22.03	Explain the role of existing libraries and packages in facilitating programmer productivity.
	22.04	Participate and contribute to a design review of a program design developed using a common program design tool (*e.g.*, UML, flowcharts, design documents, pseudocode).
	22.05	Write a program design document using standard design methodology.
	22.06	Define input and output for a program module using standard design methodology.

23.0	Solve problems using critical thinking skills, creativity and innovation. The student will be able to:
23.01	Employ critical thinking skills independently and in teams to solve problems and make decisions.
23.02	Employ critical thinking and interpersonal skills to resolve conflicts.
23.03	Identify and document workplace performance goals and monitor progress toward those goals.
23.04	Conduct technical research to gather information necessary for decision-making.
23.05	Discuss digital tools or resources to use for a real-world task based on their efficiency and effectiveness, individually and collaboratively.
24.0	Use information technology tools. The student will be able to:
24.01	Use personal information management (PIM) applications to increase workplace efficiency.
24.02	Employ technological tools to expedite workflow including word processing, databases, reports, spreadsheets, multimedia presentations, electronic calendar, contacts, email, and internet applications.
24.03	Employ computer applications to access, create, manage, integrate, and store information.
24.04	Employ collaborative/groupware applications to facilitate group work.
24.05	Use a development process in creating a computational artifact, individually and collaboratively, followed by reflection, analysis, and iteration (e.g., data-set analysis program for science and engineering fair, capstone project that includes a program, term research project based on program data).
25.0	Describe the importance of security and privacy information sharing, ownership, licensure and copyright. The student will be able to:
25.01	Describe security and privacy issues that relate to computer networks including the permanency of data on the Internet, online identity, and privacy.
25.02	Discuss the impact of government regulation on privacy and security.
25.03	Describe how different types of software licenses (e.g., open source, proprietary licenses) can be used to share and protect intellectual property.
25.04	Explain how access to information may not include the right to distribute the information.
25.05	Describe differences between open source, freeware, and proprietary software licenses, and how they apply to different types of software.
25.06	Discuss security and privacy issues that relate to computer networks.
25.07	Identify computer-related laws and analyze their impact on digital privacy, security, intellectual property, network access, contracts, and harassment.
26.0	Design a computer program to meet specific physical, operational, and interaction criteria. The student will be able to:

	26.01	Choose appropriate data types depending on the needs of the program.
	26.02	Define appropriate user prompts for clarity and usability (*e.g.*, user guidance for data ranges, data types).
	26.03	Design and develop program for efficiency (*e.g.*, less memory usage, less inputs/outputs, faster processing).
	26.04	Compare techniques for analyzing massive data collections.
	26.05	Identify the software environment required for a program to run (*e.g.*, operating system required, mobile, web-based, desktop, delivery method).
	26.06	Create mobile computing applications and/or dynamic webpages through the use of a variety of design and development tools, programming languages and mobile devices/emulators.
	26.07	Explain the role of an application programming interface (API) in the development of applications and the distinction between a programming language's syntax and the API.
	26.08	Identify the tools required to develop a program (*e.g.*, editors, compilers, linkers, integrated development environments, APIs, libraries).
	26.09	Use an industrial-strength integrated development environment to implement a program.
27.0	Create and document a computer program that uses a variety of internal and control structures for manipulating varied data types. The student will be able to:	
	27.01	Use appropriate naming conventions to define program variables and methods.
	27.02	Use a program editor to write the source code for a program.
	27.03	Write programs that use selection structures.
	27.04	Write programs that use repetition structures.
	27.05	Write programs that use nested structures.
	27.06	Use internal documentation (*e.g.*, single-line and multi-line comments, program headers, module descriptions, meaningful variable and function/module names) to document a program according to accepted standards.
	27.07	Compile, run, test and debug programs.
	27.08	Write programs that use standard arithmetic operators with different numerical data types.
	27.09	Write programs that use standard logic operators.
	27.10	Write programs that use a variety of common data types.
	27.11	Write programs that perform data conversion between standard data types.
	27.12	Write programs that define, use, search, and sort arrays.
	27.13	Write programs that use user-defined data types.

	27.14 Demonstrate understanding and use of appropriate variable scope.
	27.15 Explain recursive programming structure.
	27.16 Use global and local scope appropriately in program implementation.
28.0	Create and document an interactive computer program that employs functions, subroutines, or methods to receive, validate, and process user input. The student will be able to:
	28.01 Critically examine classical algorithms and implement an original algorithm.
	28.02 Write programs that perform user input and output.
	28.03 Write programs that validate user input (*e.g.*, range checking, data formats, valid/invalid characters).
	28.04 Write program modules such as functions, subroutines, or methods.
	28.05 Write program modules that accept arguments.
	28.06 Write program modules that return values.
	28.07 Write program modules that validate arguments and return error codes.
	28.08 Design and implement a simple simulation algorithm to analyze, represent and understand natural phenomena.
	28.09 Use APIs and libraries to facilitate programming solutions.
	28.10 Participate in a peer code review to verify program functionality, programming styles, program usability, and adherence to common programming standards.
29.0	Effectively communicate and collaborate. The student will be able to:
	29.01 Evaluate modes of communication and collaboration.
	29.02 Select appropriate tools within a project environment to communicate with project team members.
	29.03 Utilize project collaboration tools (such as version control systems and integrated development environments) while working on a collaborative software project.
	29.04 Generate, evaluate, and prioritize questions that can be researched through digital resources and online tool.
	29.05 Perform advanced searches to locate information and/or design a data-collection approach to gather original data.
	29.06 Communicate and publish key ideas and details to a variety of audiences using digital tools and media-rich resources.
30.0	Demonstrate responsible use of technology and information. The student will be able to:
	30.01 Explain the principles of cryptography by examining encryption, digital signatures, and authentication methods (e.g. explain why and how certificates are used with https for authentication and encryption).
	30.02 Implement an encryption, digital signature, or authentication method.

30.03 Describe computer security vulnerabilities and methods of attack, and evaluate their social and economic impact on computer systems and people.

Course Number: CTS0044 Occupational Completion Point: C Computer Programmer – 150 Hours – SOC Code 15-1131		
31.0	Explain key concepts that distinguish object-oriented programming from procedural programming. The student will be able to:	
	31.01	Demonstrate the understanding and use of classes, objects, attributes, and behaviors.
	31.02	Demonstrate the understanding and use of inheritance.
	31.03	Demonstrate the understanding and use of data encapsulation.
	31.04	Demonstrate the understanding and use of polymorphism.
	31.05	Use predefined functions and parameters, classes, and methods to divide a complex problem into simpler parts by using the principle of abstraction to manage complexity (e.g., by using searching and sorting as abstractions).
32.0	Create a project plan for an object-oriented programming project that defines requirements, structural design, time estimates, and testing elements. The student will be able to:	
	32.01	Write a project plan for completion of a project that includes gathering program requirements, developing the program, and testing it.
	32.02	Write a program requirements document that identifies business purpose, functional requirements, system requirements, and other common components of a requirements document.
	32.03	Design an object-oriented program using standard design methodology.
	32.04	Work with other team members to develop a project plan for a program.
	32.05	Work with other team members to write a design document for a program with multiple functions and shared data.
	32.06	Participate in design meetings that review program design documents for conformance to program requirements.
	32.07	Estimate the time to develop a program or module.
	32.08	Evaluate algorithms by their efficiency, correctness, and clarity (e.g., by analyzing and comparing execution times, testing with multiple inputs or data sets, and by debugging).
33.0	Design, document, and create object-oriented computer programs. The student will be able to:	
	33.01	Compare and contrast recursive functions to iterative methods.
	33.02	Understand the implementation of character strings in the programming language.
	33.03	Write programs that perform string processing (e.g., manipulating, comparing strings, concatenation).
	33.04	Write programs that implements user-defined data types.
	33.05	Decompose a problem by defining new functions and classes.

	33.06	Write object-oriented programs that implement inheritance.
	33.07	Write object-oriented programs that implement polymorphism.
	33.08	Develop class constructors.
	33.09	Write programs that define and use program constants.
	33.10	Write programs that perform error handling.
	33.11	Participate in program code review meetings to evaluate program code for validity, quality, performance, data integrity, and conformance to program design documents.
	33.12	Describe the concept of parallel processing as a strategy to solve large problems.
	33.13	Demonstrate concurrency by separating processes into threads of execution and dividing data into parallel streams.
	33.14	Update a program module to implement enhancements or corrections and demonstrate appropriate documentation (internal and external) related to version control.
	33.15	Write programs that use complex data structures (e.g., stacks, queues, trees, linked list).
	33.16	Write programs that are event-driven.
	33.17	Write programs that perform file input and output (i.e., sequential and random access file input/output).
	33.18	Explain intractable problems and understand that problems exists that are computationally unsolvable (undecidable) (e.g., classic intractable problems include Towers of Hanoi, TSP).
	33.19	Explain the value of heuristic algorithms to approximate solutions for intractable problems (e.g., a heuristic solution to TSP).
34.0	Design a unit test plan for an object-oriented computer program, test and debug the program, and report the results. The student will be able to:	
	34.01	Develop a test plan for an object-oriented program.
	34.02	Write test plans for event-driven programs.
	34.03	Write test plans for programs that perform file input and output.
	34.04	Perform test and debug activities on object-oriented programs, including those written by someone else.
	34.05	Perform test and debug activities on an event-driven program.
	34.06	Perform test and debug activities on programs that perform file input and output and verify the correctness of output files.
	34.07	Document the findings of testing in a test report.

35.0	Understand human interactions in intelligence. The student will be able to:	
	35.01	Describe the unique features of computers embedded in mobile devices and vehicles.
	35.02	Describe the common physical and cognitive challenges faced by users when learning to use software and hardware.
	35.03	Describe the process of designing software to support specialized forms of human-computer interaction.
	35.04	Explain the notion of intelligent behavior through computer modeling and robotics.
	35.05	Describe common measurements of machine intelligence (e.g., Turing test).
	35.06	Describe a few of the major branches of artificial intelligence (e.g., expert systems, natural language processing, machine perception, machine learning).
	35.07	Describe major applications of artificial intelligence and robotics, including, but not limited to, the medical, space, and automotive fields.

Course Number: CTS0062 Occupational Completion Point: D Database Programmer – 600 Hours – SOC Code 15-1131	
36.0	Develop an awareness of the changes taking place in the information age and how they fit into an evolving society. The student will be able to:
	36.01 Cite examples of jobs, salary, and opportunities he/she will have as a database programmer.
	36.02 Describe the role a database plays in a business.
	36.03 Understand the importance of clear communication when discussing business informational requirements.
	36.04 Identify important historical contributions in database development and design.
37.0	Develop the "big picture" of database design and how to best organize data according to business rules and/or client needs. The student will be able to:
	37.01 Identify and analyze the phases of the database development process.
	37.02 Explain what logical data modeling and database design involve.
	37.03 Compare database development process with that of the application development process.
	37.04 Distinguish between a logical model and a physical implementation.
38.0	Develop the process of creating an entity by identifying relationships. The student will be able to:
	38.01 Identify and model various types of entities.
	38.02 Identify naming and drawing conventions for entities.
	38.03 Sequence the steps that are necessary for creation of an entity.
	38.04 Analyze and model the relationships between entities.
39.0	Formulate and assemble initial entity relationship by expanding on modeling concepts. The student will be able to:
	39.01 Analyze and model attributes.
	39.02 Identify unique identifiers for each entity.
	39.03 Develop an entity relationship diagram tagging attributes with optionality.
40.0	Consider the degree and optionality of relationships of entities. The student will be able to:
	40.01 Create entity relationship models based on information requirements and interviews.
	40.02 Differentiate between one-to-many, many-to-many and one-to-one relationships.
	40.03 Identify relationship between two entities by reading a given diagram.

	40.04	Create a relationship between instances of the same entity.
	40.05	Read an entity relationship model in order to validate it.
41.0	Demonstrate proficiency in early construction stages of the data modeling process by using unique identifiers and many-to-many (M:M) relationships for building entity relationship diagrams. The student will be able to:	
	41.01	Identify the significance of an attribute that has more than one value for each entity instance.
	41.02	Evaluate appropriate methods of storing validation rules for attributes.
	41.03	Recognize unique identifiers inherited from other entities.
	41.04	Sequence the steps involved in resolving a many-to-many relationship.
42.0	Demonstrate proficiency in advanced data constructs by analyzing business requirements and diagramming entities and relationships. The student will be able to:	
	42.01	Validate that an attribute is properly placed based upon its dependence on its entity's unique identifier (UID).
	42.02	Resolve many-to-many relationships with intersection entities.
	42.03	Model advanced data constructs including recursive relationships, subtypes, and exclusive relationships.
	42.04	Create exclusive entities and relationships by using subtypes and arcs, respectively.
	42.05	Identify initial layout for presentation and generate a list of action items for members of group.
	42.06	Develop an entity relationship model using subtypes, super-types and an exclusive arc.
43.0	Demonstrate proficiency in designing and adding complexity to a logical model. The student will be able to:	
	43.01	Revise an entity relationship model according to client requirements.
	43.02	Define and give examples of hierarchical and recursive relationships.
	43.03	Differentiate between transferable and non-transferable relationships.
	43.04	Deliver a professional, formal business style presentation.
	43.05	Evaluate and critique presentation layout, design and performance.
	43.06	Construct a model using both recursion and hierarchies to express the same logical meaning.
44.0	Apply complex logical information by fine-tuning entities and the process for relating them. The student will be able to:	
	44.01	Describe a relational database and how it differs from other database systems.
	44.02	Define primary keys and foreign keys and describe their purpose.
	44.03	Describe what data integrity refers to and list some constraints.

	44.04 Explain how database design fits into the database development process.
	44.05 Translate a logical model into a relational database design.
45.0	Apply initial database design and normalization by following the set of house rules that determine how items are stored and retrieved. The student will be able to:
	45.01 Demonstrate ability to implement steps for mapping entity relationship models for implementation.
	45.02 Document an initial database design on table instance charts.
	45.03 Recognize raw data and evaluate the steps for creating a data group in unnormalized form.
46.0	Demonstrate proficiency in the technique of normalization by labeling and organizing all items in a database in such a way as to prevent any confusion or mistakes. The student will be able to:
	46.01 Differentiate between normalized and unnormalized data.
	46.02 Move data from an unnormalized form through to a third normal form.
	46.03 Demonstrate ability to test data groups for third normal form compliance.
	46.04 Identify optimized data groups from given groups of normalized data.
47.0	Demonstrate proficiency in table normalization by combining the techniques of an entity relationship model or a top-down, business approach to data with normalization or a bottom-up mathematical approach to data. The student will be able to:
	47.01 Compare the normalization and logical techniques in terms of strengths and weaknesses.
	47.02 Further define normalization and explain its benefits.
	47.03 Place tables in third normal form.
	47.04 Explain how logical data modeling rules ensure normalized tables.
	47.05 Specify referential integrity constraints and design indices.
48.0	Apply blueprint principles to begin designing a tool for creating a web-based interface access to a database. The student will be able to:
	48.01 Evaluate the transformation of business requirements into an initial layout and design for a database.
	48.02 Construct simple webpage design for personal work folder.
	48.03 Evaluate existing websites and determine quality of design.
49.0	Extend the logical model presentation model by normalizing the data and mapping the management system. The student will be able to:
	49.01 Formulate a plan of action for the Database Project using skills previously learned in this course.
	49.02 Normalize a logical model to the third normal form (3NF).
	49.03 Create a table in the database using a database authoring tool.

	49.04	Demonstrate ability to edit tables using a database authoring tool.
	49.05	Create forms that will display the table components created with a database authoring tool.
50.0	Apply techniques for building a storage management system by creating a website using templates and wizards. The student will be able to:	
	50.01	Create a website that displays the database project home.
	50.02	Link a website to create a web-enabled interface to the industry database.
	50.03	Edit the forms created and specify appropriate field labels for data entry.
51.0	Demonstrate design and functionality by constructing a group business presentation. The student will be able to:	
	51.01	Evaluate and generate criteria for a formal, business presentation.
	51.02	Construct a persuasive group presentation using the guidelines set forth in class.
52.0	Demonstrate comprehension of database modeling competency through group presentation. The student will be able to:	
	52.01	Deliver a formal business presentation for the class that discusses a logical model and initial database design.
	52.02	Demonstrate the functionality of the database and the layout/design capabilities of a database authoring tool.
	52.03	Prepare appropriate end-user documentation.
	52.04	Self-assess learning experience through the presentation and demonstration of their final database project.
53.0	Demonstrate comprehension that the database management software is a system for organizing the storage unit (or database) according to business needs and rules, through data integrity constraints. The student will be able to:	
	53.01	Identify the structural elements of a relational database table.
	53.02	List and describe the system development life cycle.
	53.03	Describe the industry implementation of the relational database management system (RDBMS) and object relational database management system (ORDBMS).
	53.04	Explain how SQL and languages that extend SQL are used in the industry product set.
	53.05	Identify the advantages of a database management system.
54.0	Demonstrate comprehension of aspects of SQL language interface by writing basic SQL statements. The student will be able to:	
	54.01	List the capabilities of SQL SELECT statements.
	54.02	Execute a basic SELECT statement.
	54.03	Differentiate between SQL statements and language commands that extend SQL.

55.0	Demonstrate proficiency working with columns, characters, and rows in SQL. The student will be able to:
55.01	Apply the concatenation operator to link columns to other columns, arithmetic expressions, or constant values to create a character expression.
55.02	Use column aliases to rename columns in the query result.
55.03	Eliminate duplicate rows in the query result.
55.04	Display the structure of a table.
55.05	Apply SQL syntax to restrict the rows returned from a query.
55.06	Demonstrate application of the WHERE clause syntax.
55.07	Construct and produce output using a SQL query containing character strings and date values.
56.0	Demonstrate proficiency in using SQL comparison operators. The student will be able to:
56.01	Apply the proper comparison operator to return a desired result.
56.02	Demonstrate proper use of BETWEEN, IN, and LIKE conditions to return a desired result.
56.03	Distinguish between zero and the value of NULL as unavailable, unassigned, unknown, or inapplicable.
56.04	Explain the use of comparison conditions and NULL.
57.0	Demonstrate proficiency in using logical comparisons and precedence rules. The student will be able to:
57.01	Evaluate logical comparisons to restrict the rows returned based on two or more conditions.
57.02	Apply the rules of precedence to determine the order in which expressions are evaluated and calculated.
57.03	Construct a query to order a results set for single or multiple columns.
57.04	Construct a query to sort a results set in ascending or descending order.
58.0	Demonstrate proficiency using SQL single row functions. The student will be able to:
58.01	Perform calculations on data.
58.02	Modify individual data items.
58.03	Use character, number and date functions in SELECT statements.
58.04	Format data and numbers for display purposes.
58.05	Convert column data types.
59.0	Demonstrate proficiency displaying data from multiple tables. The student will be able to:

	59.01	Construct SELECT statements to access data from more than one table using equity and non-equality joins.
	59.02	Use outer joins through viewing data that generally does not meet a join condition.
	59.03	Join a table to itself.
60.0	Demonstrate proficiency aggregating data using group functions. The student will be able to:	
	60.01	Identify the available group functions and describe their use.
	60.02	Demonstrate the ability to group data through the use of the GROUP BY clause.
	60.03	Demonstrate the ability to include or exclude grouped rows by using the HAVING clause.
61.0	Demonstrate proficiency utilizing subqueries. The student will be able to:	
	61.01	Write a query with an embedded subquery.
	61.02	Evaluate and perform a multiple-column subquery.
	61.03	Describe and explain the behavior of subqueries when NULL values are retrieved.
	61.04	Create a subquery in a FROM clause.
62.0	Demonstrate proficiency producing readable output with SQL language interface, reporting tool, and data manipulation language. The student will be able to:	
	62.01	Produce queries that require an input variable.
	62.02	Customize the SQL language interface and reporting environment using SET commands for control.
	62.03	Produce more readable output through the use of the column and break commands.
	62.04	Describe data manipulation language (DML) and describe various DML statements.
	62.05	Utilize data manipulation language (DML) through inserting, updating and deleting rows from a table.
	62.06	Control transactions using COMMIT and ROLLBACK statements.
63.0	Demonstrate proficiency creating and managing database objects. The student will be able to:	
	63.01	Describe the main database objects.
	63.02	Create tables and alter their definitions.
	63.03	Describe the data types that can be used when specifying column definition.
64.0	Demonstrate proficiency altering tables and constraints implementing views. The student will be able to:	
	64.01	Create, drop, rename and truncate tables using SQL.

	64.02	Identify and describe various constraints including not NULL, unique, primary key, foreign key, and check.
	64.03	Create and maintain constraints including adding, dropping, enabling, disabling, and cascading.
	64.04	Recognize views and explain how they are created, how they retrieve data and how they perform DML operations.
65.0	Demonstrate mastery of creating and implementing views, synonyms, indexes and other database objects. The student will be able to:	
	65.01	Create views, retrieve data through a view, alter the definition of a view and drop a view.
	65.02	Categorize information by using Top-N queries to retrieve specified data.
	65.03	Identify the features of a sequence and display sequence values using a data dictionary view.
	65.04	Identify the characteristics of a cached sequence.
	65.05	Modify and remove a sequence using a SQL statement.
	65.06	Identify the features of private and public synonyms.
	65.07	Identify characteristics of an index and describe different types.
	65.08	Create and remove an index using a SQL statement.
66.0	Demonstrate ability to control user access and SQL language interface and reporting tool. The student will be able to:	
	66.01	Identify the features of database security.
	66.02	Create users using SQL statements.
	66.03	Grant and revoke object privileges using a SQL language interface and reporting tool.
67.0	Demonstrate comprehension of bundling features of SQL. The student will be able to:	
	67.01	List and describe the benefits of extension languages to SQL.
	67.02	Recognize the basic SQL block and its sections.
	67.03	Declare SQL variables and describe their significance.
	67.04	Execute a SQL block.
68.0	Demonstrate comprehension working with composite data types by writing executable script files. The student will be able to:	
	68.01	Recognize the significance of the executable section and decide when to use it.
	68.02	Write statements in the executable section.
	68.03	Describe the rules of nested blocks.

	68.04 Identify and utilize appropriate coding conventions.
	68.05 Create a script that will insert, update, merge and delete data in a table.
69.0	Describe the differences between SQL and SQL extension languages. The student will be able to:
	69.01 Describe SQL extension languages.
	69.02 Differentiate between SQL and SQL extension languages.
	69.03 Explain the need for and benefits of SQL extension languages.
70.0	Create program blocks. The student will be able to:
	70.01 Describe the structure of a program block.
	70.02 Identify the different types of program blocks.
	70.03 Identify program programming environments.
	70.04 Create and execute an anonymous block.
	70.05 Output messages in program blocks.
71.0	Use variables in program blocks. The student will be able to:
	71.01 Describe how variables are used in program blocks.
	71.02 Identify the syntax for using variables.
	71.03 Declare and initialize variables.
	71.04 Assign new values to variables.
72.0	Recognize lexical units. The student will be able to:
	72.01 Describe the types of lexical units.
	72.02 Describe identifiers and identify valid and invalid identifiers.
	72.03 Describe and identify reserved words, delimiters, literals, and comments.
73.0	Recognize data types. The student will be able to:
	73.01 Describe the data type categories.
	73.02 Give examples of scalar, composite, and large object (LOB) data types.
	73.03 Identify when an object becomes eligible for garbage collection.

74.0	Use scalar data types. The student will be able to:	
	74.01	Declare and use scalar data types.
	74.02	Define guidelines for declaring and initializing variables.
75.0	Use various types of joins. The student will be able to:	
	75.01	Construct and execute SELECT statements using an equijoin.
	75.02	Construct and execute SELECT statements using a non-equijoin.
	75.03	Construct and execute SELECT statements using an outer join.
	75.04	Construct and execute SELECT statements that result in cross join.
76.0	Use SQL group functions and subqueries. The student will be able to:	
	76.01	Construct and execute an SQL query using group functions to determine a sum total, an average amount, and a maximum value.
	76.02	Construct and execute an SQL query that groups data based on specified criteria.
	76.03	Construct and execute an SQL query that contains a WHERE clause using a single-row subquery.
	76.04	Construct and execute an SQL query that contains a WHERE clause using a multiple-row subquery.
77.0	Write executable statements. The student will be able to:	
	77.01	Construct variable assignment statements.
	77.02	Construct statements using built-in SQL functions.
	77.03	Differentiate between implicit and explicit data type conversions.
	77.04	Describe when implicit data type conversions take place.
	77.05	List the drawbacks of implicit data type conversions.
	77.06	Construct statements using functions to explicitly convert data types.
	77.07	Construct statements using operators.
78.0	Use nested blocks and variable scope. The student will be able to:	
	78.01	Understand the scope and visibility of variables.
	78.02	Write nested blocks and qualify variables with labels.
	78.03	Describe the scope of an exception.

	78.04	Describe the effect of exception propagation in nested blocks.
79.0	Use good programming practices. The student will be able to:	
	79.01	List examples of good programming practices.
	79.02	Insert comments into code.
	79.03	Follow formatting guidelines when writing code.
80.0	Write DML statements to manipulate data. The student will be able to:	
	80.01	Construct and execute a statement to insert data into a table.
	80.02	Construct and execute a statement to update data in a table.
	80.03	Construct and execute a statement to delete data from a table.
	80.04	Construct and execute a statement to merge data into a table.
81.0	Retrieve data. The student will be able to:	
	81.01	Identify SQL statements that can be directly included in an executable block.
	81.02	Construct and execute an INTO clause to hold values returned by a single-row SELECT statement.
	81.03	Construct statements that retrieve data.
82.0	Manipulate data. The student will be able to:	
	82.01	Describe when to use implicit or explicit cursors.
	82.02	Create code to use SQL implicit cursor attributes to evaluate cursor activity.
83.0	Use transaction control statements. The student will be able to:	
	83.01	Define a transaction and give an example.
	83.02	Construct and execute a transaction control statement.
84.0	Use IF conditional control statements. The student will be able to:	
	84.01	Construct and use an IF statement.
	84.02	Construct and use an IF -ELSIF statement.
	84.03	Create control statements to handle NULL conditions in an IF statement.
85.0	Use CASE conditional control statements. The student will be able to:	

	85.01	Construct and use CASE statements.
	85.02	Construct and use CASE expressions.
	85.03	Include syntax to handle NULL conditions in a CASE statement.
	85.04	Include syntax to handle Boolean conditions in IF and CASE statements.
86.0	Use basic LOOP iterative control statements. The student will be able to:	
	86.01	Describe the types of LOOP statements and their uses.
	86.02	Create a program containing a basic loop and an EXIT statement.
	86.03	Create a program containing a basic loop and an EXIT statement with conditional termination.
87.0	Use WHILE and FOR loop iterative control statements. The student will be able to:	
	87.01	Construct and use the WHILE looping construct.
	87.02	Construct and use the FOR looping construct.
	87.03	Describe when a WHILE loop is used.
	87.04	Describe when a FOR loop is used.
88.0	Use nested loop iterative control statements. The student will be able to:	
	88.01	Construct and execute a program using nested loops.
	88.02	Evaluate a nested loop construct and identify the exit point.
89.0	Use explicit cursors. The student will be able to:	
	89.01	List the guidelines for declaring and controlling explicit cursors.
	89.02	Create code to open a cursor and fetch a piece of data into a variable.
	89.03	Use a simple loop to fetch multiple rows from a cursor.
	89.04	Create code to close a cursor.
90.0	Use explicit cursor attributes. The student will be able to:	
	90.01	Define a record structure.
	90.02	Create code to process the row of an active set using record types in cursors.
	90.03	Use cursor attributes to retrieve information about the state of an explicit cursor.

91.0	Use cursor FOR loops. The student will be able to:
	91.01 List and explain the benefits of using a cursor FOR loops.
	91.02 Create code to declare a cursor and manipulate it in a FOR loop.
	91.03 Create code containing a cursor FOR loop using a subquery.
92.0	Use cursors with parameters. The student will be able to:
	92.01 List the benefits of using parameters with cursors.
	92.02 Create code to declare and manipulate a cursor with a parameter.
93.0	Use cursors for update transactions. The student will be able to:
	93.01 Create code to lock rows before an update using the appropriate clause.
	93.02 Explain the effect of using NOWAIT in an update cursor declaration.
	93.03 Create code to use the current row of the cursor in an UPDATE or DELETE statement.
94.0	Use multiple cursors. The student will be able to:
	94.01 Explain the need for using multiple cursors to produce multilevel reports.
	94.02 Create code to declare and manipulate multiple cursors within nested loops.
	94.03 Create code to declare and manipulate multiple cursors using parameters.
95.0	Handle exceptions. The student will be able to:
	95.01 Describe the advantages of including exception handling code.
	95.02 Describe the purpose of an EXCEPTION section in a program block.
	95.03 Create code to include an EXCEPTION section.
	95.04 List the guidelines for exception handling.
96.0	Trap server exceptions. The student will be able to:
	96.01 Distinguish between errors defined by the server and those defined by the programmer.
	96.02 Differentiate between errors that are handled implicitly and explicitly by the server.
	96.03 Write code to trap a predefined server error.
	96.04 Write code to trap a non-predefined server error.

	96.05	Write code to identify an exception by error code and by error message.
97.0	Trap user-defined exceptions. The student will be able to:	
	97.01	Write code to name a user-defined exception.
	97.02	Write code to raise an exception.
	97.03	Write code to handle a raised exception.
98.0	Create procedures. The student will be able to:	
	98.01	Differentiate between anonymous blocks and subprograms.
	98.02	Identify the benefits of using subprograms.
	98.03	Describe a stored procedure.
	98.04	Create a procedure.
	98.05	Describe how a stored procedure is invoked.
99.0	Use parameters in procedures. The student will be able to:	
	99.01	Describe how parameters contribute to a procedure.
	99.02	Define a parameter.
	99.03	Create a procedure using a parameter.
	99.04	Invoke a procedure that has parameters.
	99.05	Distinguish between formal and actual parameters.
100.0	Pass parameters. The student will be able to:	
	100.01	List the types of parameter modes.
	100.02	Create a procedure that passes parameters.
	100.03	Identify methods for passing parameters.
	100.04	Describe the default option for parameters.
101.0	Create stored functions. The student will be able to:	
	101.01	Describe the difference between a stored procedure and a stored function.

	101.02 Create a program block containing a function.
	101.03 Identify ways in which functions may be invoked.
	101.04 Create a program block that invokes a function that has parameters.
102.0	Use functions in SQL statements. The student will be able to:
	102.01 Describe where user-defined functions can be called from within an SQL statement.
	102.02 Describe the restrictions on calling functions from SQL statements.
	102.03 Describe the purpose of the Data Dictionary.
	102.04 Differentiate different types of Data Dictionary views.
	102.05 Write SQL SELECT statements to retrieve information from the Data Dictionary.
103.0	Manage procedures and functions. The student will be able to:
	103.01 Describe how exceptions are propagated.
	103.02 Remove a function and a procedure.
	103.03 Use Data Dictionary views to identify and manage stored procedures.
104.0	Manage object privileges. The student will be able to:
	104.01 List and explain several object privileges.
	104.02 Explain the function of the EXECUTE object privilege.
	104.03 Write SQL statements to grant and revoke object privileges.
105.0	Use invoker’s rights. The student will be able to:
	105.01 Contrast invoker’s rights with definer’s rights.
	105.02 Create a procedure that uses invoker’s rights.
106.0	Create packages. The student will be able to:
	106.01 Describe a package, its components, and the reasons for use.
	106.02 Create packages containing related variables, cursors, constants, exceptions, procedures, and functions.
	106.03 Create a program block that invokes a package construct.

107.0	Manage package constructs. The student will be able to:
	107.01 Explain the difference between public and private package constructs.
	107.02 Designate a package construct as either public or private.
	107.03 Specify the syntax to drop a package.
	107.04 Identify Data Dictionary views used to manage packages.
	107.05 Identify the guidelines for using packages.
108.0	Use advanced package concepts. The student will be able to:
	108.01 Write packages that use the overloading feature.
	108.02 Write packages that use forward declarations.
	108.03 Explain the purpose of a package initialization block.
	108.04 Identify restrictions on using packaged functions in SQL statements.
109.0	Manage persistent state of package variables. The student will be able to:
	109.01 Identify persistent states of package variables.
	109.02 Control the persistent state of a package cursor.
110.0	Use vendor-supplied packages. The student will be able to:
	110.01 Describe common uses for vendor-supplied packages.
	110.02 Use the syntax to specify messages for a vendor-supplied package.
	110.03 Identify the exceptions used in conjunction with vendor-supplied packages.
111.0	Understand dynamic SQL. The student will be able to:
	111.01 Identify the stages through which all SQL statements pass.
	111.02 Describe the reasons for using dynamic SQL to create an SQL statement.
	111.03 List statements supporting Native Dynamic SQL.
112.0	Understand triggers. The student will be able to:
	112.01 Describe database triggers and their uses.
	112.02 Differentiate between a database trigger and an application trigger.

	112.03 List the guidelines for using triggers.
	112.04 Compare and contrast database triggers and stored procedures.
113.0	Create DML triggers. The student will be able to:
	113.01 Create a DML trigger and identify its components.
	113.02 Create a statement level trigger.
	113.03 Describe the trigger firing sequence options.
	113.04 Create a DML trigger that uses conditional predicates.
	113.05 Create a row level trigger.
	113.06 Create a row level trigger that uses OLD and NEW qualifiers.
	113.07 Create an INSTEAD OF trigger.
114.0	Create DDL and database event triggers. The student will be able to:
	114.01 Describe the events that cause DDL and database event triggers to fire.
	114.02 Create a trigger for a DDL statement.
	114.03 Create a trigger for a database event.
	114.04 Describe the functionality of the CALL statement.
	114.05 Describe the cause of a mutating table.
115.0	Manage triggers. The student will be able to:
	115.01 View trigger information in the Data Dictionary.
	115.02 Disable and enable a database trigger.
	115.03 Remove a trigger from the database.
116.0	Use large object data types. The student will be able to:
	116.01 Compare and contrast LONG and LOB data types.
	116.02 Describe LOB data types and how they are used.
	116.03 Differentiate between internal and external LOBs.
	116.04 Create and maintain LOB data types.

	116.05 Migrate data from LONG to LOB.
117.0	Manage binary types. The student will be able to:
	117.01 Define binary column data type.
	117.02 Create directory objects and view them in the Data Dictionary.
	117.03 Manage and manipulate binary types.
118.0	Manage indexes. The student will be able to:
	118.01 Create and manipulate user-defined records.
	118.02 Create an index.
	118.03 Describe the difference between records, tables, and indexes.
119.0	Manage dependencies. The student will be able to:
	119.01 Describe the implications of procedural dependencies.
	119.02 Contrast dependent objects and referenced objects.
	119.03 View dependency information in the Data Dictionary.
	119.04 Use a script to create the objects required to display dependencies.
	119.05 Use views to display dependencies.
	119.06 Describe how to minimize dependency failures.
120.0	Demonstrate an understanding of Agile Development. The student will be able to:
	120.01 Compare Agile project development to the waterfall approach.
	120.02 Describe the Agile manifesto and the 12 principles.
	120.03 Describe the benefits of Agile development.
121.0	Program a database application. The student will be able to:
	121.01 Utilize loop statements.
	121.02 Given a scenario, use arithmetic, comparison, and pattern-matching operators.
	121.03 Create user-defined functions.
	121.04 Utilize common built-in functions.

	121.05 Declare variables in modules and procedures.
	121.06 Declare arrays, and initialize elements of arrays.
	121.07 Declare and use object variables and collections, and use their associated properties and methods.
	121.08 Declare symbolic constants, and make them available locally or publicly.
	121.09 Respond to events.
122.0	Utilize the basic concepts of database design. The student will be able to:
	122.01 Apply basic concepts of normalization.
	122.02 Utilize the cascade update and cascade delete options.
123.0	Utilize SQL and union queries. The student will be able to:
	123.01 Utilize SQL to write common queries.
	123.02 Refer to objects by using SQL.
	123.03 Utilize union queries.
124.0	Implement program statements using objects. The student will be able to:
	124.01 Determine when to use data access objects.
	124.02 Differentiate between objects and collections.
	124.03 Write statements that access and modify database objects.
	124.04 Utilize data access objects.
	124.05 Select appropriate methods and property settings for use with specified objects.
125.0	Utilize debugging tools and write error handlers. The student will be able to:
	125.01 Trap errors.
	125.02 Utilize debugging tools to suspend program execution, and to examine, step through, and reset execution of code.
	125.03 Debug code samples.
	125.04 Utilize the Debugger to monitor variable values.
	125.05 Write an error handler.
126.0	Demonstrate file I/O. The student will be able to:

	126.01 Read from files.
	126.02 Write to files.
	126.03 Utilize record locking.
127.0	Create forms and identify all the properties of a form. The student will be able to:
	127.01 Choose form-specific and report-specific properties to set.
	127.02 Choose control properties to set.
	127.03 Assign event-handling procedures to controls in a form.
	127.04 Define and create form and report modules.
	127.05 Identify the scope of a form or report module.
	127.06 Open multiple instances of a form, and refer to them.
	127.07 Assign values to form properties.
	127.08 Use form methods.
128.0	Manipulate data using object models. The student will be able to:
	128.01 Connect to a data source.
	128.02 Open a recordset.
	128.03 Insert, update, merge and delete data.
129.0	Develop custom controls. The student will be able to:
	129.01 Set properties for custom controls.
	129.02 Customize user interface controls.
130.0	Utilize API functions. The student will be able to:
	130.01 Properly declare functions.
	130.02 Use the by value and by reference parameters.
131.0	Demonstrate and implement database replication using programming tools. The student will be able to:
	131.01 Make a database replicable.
	131.02 View a synchronization schedule.

	131.03 Explain how synchronization conflicts are resolved.
	131.04 Identify the advantages of using replication of synchronization.
	131.05 Identify the changes that the database engine makes when it converts a nonreplicable database into replicable database.
132.0	Analyze and implement security options. The student will be able to:
	132.01 Analyze a scenario, and recommend an appropriate type of security.
	132.02 Explain the steps for implementing security.
	132.03 Analyze code to ensure that it sets security options.
	132.04 Write code to implement security options.
133.0	Implement client/server applications. The student will be able to:
	133.01 Demonstrate SQL pass through queries and application queries.
	133.02 Access external data.
	133.03 Trap errors that are generated by the server.
	133.04 Optimize connections.
	133.05 Optimize performance for a given client/server application.
134.0	Optimize the performance of a database. The student will be able to:
	134.01 Differentiate between single-field and multiple-field indexes.
	134.02 Optimize queries.
	134.03 Restructure queries to allow faster execution.
	134.04 Optimize performance in distributed applications.
	134.05 Optimize performance for client/server applications.
135.0	Perform application distribution. The student will be able to:
	135.01 Prepare an application for distribution.
	135.02 Analyze various methods to distribute a client/server application.
	135.03 Distribute custom controls with an application.
	135.04 Provide online help.

136.0	Test and debug databases. The student will be able to:
	136.01 Implement error handling.
	136.02 Test and debug library databases.
137.0	Describe the difference between relational and NoSQL databases. The student will be able to:
	137.01 Describe the advantages and disadvantages of NoSQL databases.
	137.02 Describe the types of NoSQL databases (e.g., key-value store, column-based, graph-based, document-based).
	137.03 Describe when a NoSQL database should be used for storage.
138.0	Demonstrate an understanding of Data Science and the concept of Data mining. The student will be able to:
	138.01 Define Data Science.
	138.02 Define Data Mining.
	138.03 Describe and compare Structured Data and Non-Structured Data.
	138.04 Describe and model the Data Science Life Cycle.
	138.05 Describe and compare various Deep Learning Frameworks available to Data Science.
	138.06 Describe and compare Data Science and Data Analytics.

Additional Information

Laboratory Activities

Laboratory investigations that include scientific inquiry, research, measurement, problem solving, emerging technologies, tools and equipment, as well as, experimental, quality, and safety procedures are an integral part of this career and technical program/course. Laboratory investigations benefit all students by developing an understanding of the complexity and ambiguity of empirical work, as well as the skills required to manage, operate, calibrate and troubleshoot equipment/tools used to make observations. Students understand measurement error; and have the skills to aggregate, interpret, and present the resulting data. Equipment and supplies should be provided to enhance hands-on experiences for students.

Special Notes

MyCareerShines is an interactive resource to assist students in identifying their ideal career and to enhance preparation for employment. Teachers are encouraged to integrate this resource into the program curriculum to meet the employability goals for each student. Access MyCareerShines by visiting: www.mycareershines.org.

Career and Technical Student Organization (CTSO)

Phi Beta Lambda and Business Professionals of America (BPA) are the intercurricular student organizations providing leadership training and reinforcing specific career and technical skills. Career and Technical Student Organizations provide activities for students as an integral part of the instruction offered.

Cooperative Training – OJT

On-the-job training is appropriate but not required for this program. Whenever offered, the rules, guidelines, and requirements specified in the OJT framework apply.

Basic Skills

In a Career Certificate Program offered for 450 hours or more, in accordance with Rule 6A-10.040, F.A.C., the minimum basic skills grade levels required for postsecondary adult career and technical students to complete this program are: Mathematics 9, Language 9, and Reading 9. These grade level numbers correspond to a grade equivalent score obtained on a state designated basic skills examination.

Adult students with disabilities, as defined in Section 1004.02(7), Florida Statutes, may be exempted from meeting the Basic Skills requirements (Rule 6A-10.040). Students served in exceptional student education (except gifted) as defined in s. 1003.01(3)(a), F.S., may also be exempted from meeting the Basic Skills requirement. Each school district and Florida College must adopt a policy addressing procedures for exempting eligible students with disabilities from the Basic Skills requirement as permitted in Section 1004.91(3), F.S.

Students who possess a college degree at the Associate of Applied Science level or higher; who have completed or are exempt from the college entry-level examination; or who have passed a state, national, or industry licensure exam are exempt from meeting the Basic Skills requirement (Rule 6A-10.040, F.A.C.) Exemptions from state, national or industry licensure are limited to the certifications listed on the Basic Skills and Licensure Exemption List which may be accessed from the CTE Program Resources page.

Accommodations

Federal and state legislation requires the provision of accommodations for students with disabilities to meet individual needs and ensure equal access. Postsecondary students with disabilities must self-identify, present documentation, request accommodations if needed, and develop a plan with their counselor and/or instructors. Accommodations received in postsecondary education may differ from those received in secondary education. Accommodations change the way the student is instructed. Students with disabilities may need accommodations in such areas as instructional methods and materials, assignments and assessments, time demands and schedules, learning environment, assistive technology and special communication systems. Documentation of the accommodations requested and provided should be maintained in a confidential file.

Note: postsecondary curriculum and regulated secondary programs cannot be modified.

Additional Resources

For additional information regarding articulation agreements, Bright Futures Scholarships, Fine Arts/Practical Arts Credit and Equivalent Mathematics and Equally Rigorous Science Courses please refer to:
http://www.fldoe.org/academics/career-adult-edu/career-tech-edu/program-resources.stml